

Requested Patent: JP2001067348A

Title:

METHOD AND DEVICE FOR COMPRESSING STRUCTURED DOCUMENTS AND COMPUTER-READABLE RECORDING MEDIUM RECORDING STRUCTURED DOCUMENT COMPRESSING PROGRAM ;

Abstracted Patent: JP2001067348 ;

Publication Date: 2001-03-16 ;

Inventor(s): YAHAGI HIRONORI ;

Applicant(s): FUJITSU LTD ;

Application Number: JP20000098043 20000331 ;

Priority Number(s): ;

IPC Classification: G06F17/21; G06F12/00; G06F17/30; H03M7/30 ;

Equivalents: ;

ABSTRACT:

PROBLEM TO BE SOLVED: To improve compressibility by analyzing the tree structure of an element in a document realizing value and shifting information on a leaf element into a starting tag as the attribute of a master element to maintain the feature of a structured document to make compression of a tag part possible. SOLUTION: A document realizing value analyzing part 20 analyzes a document realizing value forming an XML document and outputs the element list (file) of leaves as the analyzing result of the tree structure (master and slave relation) of an element. At this time, in the list of the elements of the leaves outputted from the part 20, elements arranged as the leaves of the tree structure without having slave elements are detected and a corresponding relation between the element of the leaves and a master element is clearly written. A document realizing value changing part moves information on the leaf element at the document realizing value into the starting tag of the master element as the attribute of the master element of the leaf element in accordance with the analyzing result (the list of the elements of the leaves) by the part 20 so as to simplify the expression of the document realizing value. Then, the XML document is outputted and stored in a document storage part 10, etc., after being changed and compressed.

(19)日本国特許庁 (JP)

(12) 公開特許公報 (A)

(11)特許出願番号

特開2001-67348

(P2001-67348A)

(43)公開日 平成13年3月16日 (2001.3.16)

(51)Int.Cl. ¹	識別記号	F 1	テ-レコ-ド(参考)
G 0 6 F 17/21	5 0 1	G 0 6 F 17/21	5 0 1 T 5 B 0 0 9
12/00	5 1 1	12/00	5 1 1 A 5 B 0 7 5
	5 4 6		5 4 6 A 5 B 0 8 2
17/30	1 7 0	17/30	1 7 0 A 5 J 0 6 4
	2 3 0		2 3 0 A

審査請求 未請求 請求項の数28 O L (全 53 頁) 最終頁に続く

(21)出願番号	特願2000-98043(P2000-98043)	(71)出願人	000005223 富士通株式会社 神奈川県川崎市中原区上小田中4丁目1番 1号
(22)出願日	平成12年3月31日 (2000.3.31)	(72)発明者	矢作 裕紀 神奈川県川崎市中原区上小田中4丁目1番 1号 富士通株式会社内
(31)優先権主張番号	特願平11-173468	(74)代理人	100092978 弁理士 真田 有
(32)優先日	平成11年6月21日 (1999.6.21)		
(33)優先権主張国	日本 (JP)		

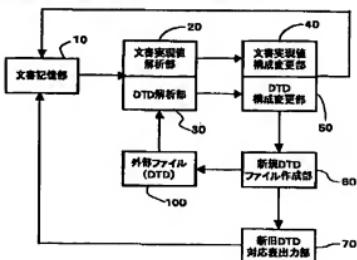
最終頁に続く

(54)【発明の名称】 構造化文書の圧縮方法および圧縮装置並びに構造化文書圧縮プログラムを記録したコンピュータ可収可能な記録媒体

(57)【要約】

【課題】 構造化文書の特徴を損なうことなくタグ部分の圧縮を可能にし、構造化文書の圧縮率の向上をはかる。

【解決手段】 構造化文書を成す文書実現値における要素の木構造を解析する文書実現値解析部20と、この文書実現値解析部20による解析結果に従い、木構造の葉要素についての情報を、その葉要素の親要素の属性として該親要素の開始タグ内に移す文書実現値構成変更部40とをそなえて構成する。



【特許請求の範囲】

【請求項1】 構造化文書を圧縮する方法であって、該構造化文書を成す文書実現値における要素の木構造を解析する文書実現値解析ステップと、該文書実現値解析ステップでの解析結果に従い、該木構造の葉となる要素（以下、葉要素という）についての情報を、該葉要素の親要素の属性として該親要素の開始タグ内に移す文書実現値構成変更ステップとを有することを特徴とする、構造化文書の圧縮方法。

【請求項2】 該文書実現値構成変更ステップにおいて、該葉要素についての開始タグ、終了タグおよび内容を該文書実現値から削除し、該葉要素についての情報である要素名および内容を、それぞれ該親要素の属性名および属性値として該親要素の開始タグ内に付加することを特徴とする、請求項1記載の構造化文書の圧縮方法。

【請求項3】 該葉要素の開始タグ内に該葉要素についての情報である属性が記述されている場合、該文書実現値構成変更ステップにおいて、該属性にかかる属性名および属性値を、それぞれ該親要素の属性名および属性値として該親要素の開始タグ内に付加することを特徴とする、請求項2記載の構造化文書の圧縮方法。

【請求項4】 該文書実現値構成変更ステップにおいて、該親要素の終了タグを削除するとともに、該親要素の開始タグを空要素タグに変更することを特徴とする、請求項1～請求項3のいずれか1項に記載の構造化文書の圧縮方法。

【請求項5】 該構造化文書を成す文書型定義における要素の木構造を解析する文書型定義解析ステップと、該文書型定義解析ステップでの解析結果に従い、該木構造の葉となる要素（以下、葉要素という）についての情報を、該文書型定義から削除し、該葉要素の親要素の属性として該文書型定義で再定義する文書型定義構成変更ステップとをさらに有することを特徴とする、請求項1～請求項4のいずれか1項に記載の構造化文書の圧縮方法。

【請求項6】 該文書型定義構成変更ステップにおいて、該葉要素の要素型宣言を該文書型定義から削除するとともに該葉要素にかかる記述を該親要素の要素型宣言から削除し、該葉要素の要素型宣言にかかる情報を、該親要素の属性として該親要素の属性リスト宣言で再定義することを特徴とする、請求項5記載の構造化文書の圧縮方法。

【請求項7】 該文書型定義で該葉要素の属性が該葉要素の属性リスト宣言により定義されている場合、該文書型定義構成変更ステップにおいて、該葉要素の属性リスト宣言を該文書型定義から削除し、該葉要素の属性を、該親要素の属性として該親要素の属性リスト宣言で再定義することを特徴とする、請求項6記載の構造化文書の圧縮方法。

【請求項8】 構造化文書を圧縮する方法であって、

該構造化文書を成す文書実現値のタグ内の記述を解析する文書実現値解析ステップと、

該文書実現値解析ステップでの解析結果に従って、該文書実現値のタグ内に記述された文字列と該文字列よりも短く且つ該文字列を特定しうる短縮文字列とを対応させるタグ辞書を作成するタグ辞書作成ステップと、該タグ辞書作成ステップで作成された該タグ辞書を用いて、該文書実現値のタグ内に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書実現値文字列置換ステップとを有することを特徴とする、構造化文書の圧縮方法。

【請求項9】 構造化文書を圧縮する方法であって、該構造化文書を成す文書実現値のタグ内の記述を解析する文書実現値解析ステップと、

該構造化文書を成す文書型定義の記述を解析する文書型定義解析ステップと、

該文書実現値解析ステップおよび該文書型定義解析ステップでの解析結果に従って、該文書実現値のタグ内および該文書型定義に記述された文字列と該文字列よりも短く且つ該文字列を特定しうる短縮文字列とを対応させるタグ辞書を作成するタグ辞書作成ステップと、

該タグ辞書作成ステップで作成された該タグ辞書を用いて、該文書実現値のタグ内に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書実現値文字列置換ステップと。

該タグ辞書作成ステップで作成された該タグ辞書を用いて、該文書型定義に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書型定義文字列置換ステップとを有することを特徴とする、構造化文書の圧縮方法。

【請求項10】 該タグ内もしくは該文書型定義に記述された要素名および属性名を前記文字列として扱い、該要素名および該属性名を該短縮文字列に置き換えることを特徴とする、請求項8または請求項9に記載の構造化文書の圧縮方法。

【請求項11】 単語文字列と該単語文字列よりも短く且つ該単語文字列を特定しうる短縮文字列とを対応させる単語辞書を用いて、該文書実現値の内容に含まれる単語文字列を、当該単語文字列に対応する短縮文字列に置き換える単語文字列置換ステップを有することを特徴とする、請求項8～請求項10のいずれか1項に記載の構造化文書の圧縮方法。

【請求項12】 該タグ内もしくは該文書型定義に記述された文字列を該短縮文字列に置き換えるとともに該単語文字列を該短縮文字列に置き換えた後に、これらの文字列を可変長符号化により圧縮する可変長符号化ステップを有することを特徴とする、請求項11記載の構造化文書の圧縮方法。

【請求項13】 構造化文書を圧縮する装置であって、該構造化文書を成す文書実現値における要素の木構造を

解析する文書実現値解析部と、該文書実現値解析部による解析結果に従い、該木構造の葉となる要素（以下、葉要素という）についての情報を、該葉要素の親要素の属性として該親要素の開始タグ内に移す文書実現値構成変更部とをそなえて構成されたことを特徴とする、構造化文書の圧縮装置。

【請求項14】 該文書実現値構成部が、該葉要素についての開始タグ、終了タグおよび内容を該文書実現値から削除し、該葉要素についての情報である要素名および内容を、それぞれ該親要素の属性名および属性値として該親要素の開始タグ内に付加することを特徴とする、請求項13記載の構造化文書の圧縮装置。

【請求項15】 該葉要素の開始タグ内に該葉要素についての情報である属性名が記述されている場合、該文書実現値構成変更部が、該属性にかかる属性名および属性値を、それぞれ該親要素の属性名および属性値として該親要素の開始タグ内に付加することを特徴とする、請求項14記載の構造化文書の圧縮装置。

【請求項16】 該文書実現値構成変更部が、該親要素の終了タグを削除とともに、該葉要素の開始タグを空要素タグに変更することを特徴とする、請求項13～請求項15のいずれか1項に記載の構造化文書の圧縮装置。

【請求項17】 該構造化文書を成す文書型定義における要素の木構造を解析する文書型定義解析部と、該文書型定義解析部による解析結果に従い、該木構造の葉となる要素（以下、葉要素という）についての情報を、該文書型定義から削除し、該葉要素の親要素の属性として該文書型定義で再定義する文書型定義構成変更部とをさらにそなえたことを特徴とする、請求項13～請求項16のいずれか1項に記載の構造化文書の圧縮装置。

【請求項18】 構造化文書を圧縮する装置であって、該構造化文書を成す文書実現値のタグ内の記述を解析する文書実現値解析部と、

該文書実現値解析部による解析結果に従って、該文書実現値のタグ内に記述された文字列と該文字列よりも短く且つ該文字列を特定しうる短縮文字列とを対応させるタグ辞書を作成するタグ辞書作成部と、

該タグ辞書作成部により作成された該タグ辞書を用いて、該文書実現値のタグ内に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書実現値文字列置換部とそなえて構成されたことを特徴とする、構造化文書の圧縮装置。

【請求項19】 構造化文書を圧縮する装置であって、該構造化文書を成す文書実現値のタグ内の記述を解析する文書実現値解析部と、

該構造化文書を成す文書型定義の記述を解析する文書型定義解析部と、

該文書実現値解析部および該文書型定義解析部による解

析結果に従って、該文書実現値のタグ内および該文書型定義に記述された文字列と該文字列よりも短く且つ該文字列を特定しうる短縮文字列とを対応させるタグ辞書を作成するタグ辞書作成部と、

該タグ辞書作成部により作成された該タグ辞書を用いて、該文書実現値のタグ内に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書実現値文字列置換部と、

該タグ辞書作成部により作成された該タグ辞書を用いて、該文書型定義に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書型定義文字列置換部とをそなえて構成されたことを特徴とする、構造化文書の圧縮装置。

【請求項20】 該タグ内もしくは該文書型定義に記述された要素名および属性名を前記文字列として扱い、該要素名および属性名を該短縮文字列に置き換えることを特徴とする、請求項18または請求項19に記載の構造化文書の圧縮装置。

【請求項21】 構造化文書を圧縮する機能をコンピュータにより実現するための構造化文書圧縮プログラムを格納したコンピュータ読取可能な記録媒体であって、該構造化文書圧縮プログラムが、

該構造化文書を成す文書実現値における要素の木構造を解析する文書実現値解析部、および、該文書実現値解析部による解析結果に従い、該木構造の葉となる要素（以下、葉要素という）についての情報を、該葉要素の親要素の属性として該親要素の開始タグ内に移す文書実現値構成変更部として、該コンピュータを機能させることを特徴とする、構造化文書圧縮プログラムを格納したコンピュータ読取可能な記録媒体。

【請求項22】 該構造化文書圧縮プログラムが、該文書実現値構成部により、該葉要素についての開始タグ、終了タグおよび内容を該文書実現値から削除し、該葉要素についての情報である要素名および内容を、それぞれ該親要素の属性名および属性値として該親要素の開始タグ内に付加するよう、該コンピュータを機能させることを特徴とする、請求項21記載の構造化文書圧縮プログラムを格納したコンピュータ読取可能な記録媒体。

【請求項23】 該構造化文書圧縮プログラムが、該葉要素の開始タグ内に該葉要素についての情報である属性が記述されている場合、該文書実現値構成変更部により、該属性にかかる属性名および属性値を、それぞれ該親要素の属性名および属性値として該親要素の開始タグ内に付加するよう、該コンピュータを機能させることを特徴とする、請求項22記載の構造化文書圧縮プログラムを格納したコンピュータ読取可能な記録媒体。

【請求項24】 該構造化文書圧縮プログラムが、該文書実現値構成変更部により、該親要素の終了タグを削除するとともに該親要素の開始タグを空要素タグに変更するよう、該コンピュータを機能させることを特徴とす

る、請求項21～請求項23のいずれか1項に記載の構造化文書圧縮プログラムを格納したコンピュータ読取可能な記録媒体。

【請求項25】 該構造化文書圧縮プログラムが、さらに、

該構造化文書を成す文書型定義における要素の木構造を解析する文書型定義解析部、および、

該文書型定義解析部による解析結果に従い、該木構造の葉となる要素（以下、葉要素という）についての情報を、該文書型定義から削除し、該は要素の親要素の属性として該文書型定義で再定義する文書型定義精変更部として、該コンピュータを機能させることを特徴とする、請求項21～請求項24のいずれか1項に記載の構造化文書圧縮プログラムを格納したコンピュータ読取可能な記録媒体。

【請求項26】 構造化文書を圧縮する機能をコンピュータにより実現するための構造化文書圧縮プログラムを格納したコンピュータ読取可能な記録媒体であって、該構造化文書圧縮プログラムが、

該構造化文書を成す文書実現値のタグ内の記述を解析する文書実現値解析部、

該文書実現値解析部による解析結果に従って、該文書実現値のタグ内に記述された文字列と該文字列よりも短く且つ該文字列を特定しうる短縮文字列とを対応させるタグ辞書を作成するタグ辞書作成部、および、

該タグ辞書作成部により作成された該タグ辞書を用いて、該文書実現値のタグ内に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書実現値文字列置換部として、該コンピュータを機能させることを特徴とする、構造化文書圧縮プログラムを格納したコンピュータ読取可能な記録媒体。

【請求項27】 構造化文書を圧縮する機能をコンピュータにより実現するための構造化文書圧縮プログラムを格納したコンピュータ読取可能な記録媒体であって、該構造化文書圧縮プログラムが、

該構造化文書を成す文書実現値のタグ内の記述を解析する文書実現値解析部、

該構造化文書を成す文書型定義の記述を解析する文書型定義解析部、

該文書実現値解析部および該文書型定義解析部による解析結果に従って、該文書実現値のタグ内および該文書型定義に記述された文字列と該文字列よりも短く且つ該文字列を特定しうる短縮文字列とを対応させるタグ辞書を作成するタグ辞書作成部、

該タグ辞書作成部により作成された該タグ辞書を用いて、該文書実現値のタグ内に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書実現値文字列置換部、および、

該タグ辞書作成部により作成された該タグ辞書を用いて、該文書型定義に記述された文字列を、当該文字列に

対応する短縮文字列に置き換える文書型定義文字列置換部として、該コンピュータを機能させることを特徴とする、構造化文書圧縮プログラムを格納したコンピュータ読取可能な記録媒体。

【請求項28】 該構造化文書圧縮プログラムが、該タグ内もしくは該文書型定義に記述された要素名および属性名を前記文字列として扱い、該コンピュータに、該要素名および該属性名を該短縮文字列に置換させることを特徴とする、請求項26または請求項27に記載の構造化文書圧縮プログラムを格納したコンピュータ読取可能な記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術】本発明は、HTML (Hyper Text Markup Language), SGML (Standard Generalized Markup Language) や XML (Extensible Markup Language) 等の構造化文書を圧縮するための、方法および装置、並びに、プログラムを記録したコンピュータ読取可能な記録媒体に関する。

【0002】

【従来の技術】近年、計算機やインターネットやインターネットの普及に伴い、文書、ソフトウェア、数値、画像データ等の様々な種類のデータを含む電子文書が増加している。そして、電子文書のデータ量が大きい場合には、電子文書データから冗長な部分を省いてそのデータを圧縮することにより、メモリに記憶される電子文書のデータ量を減らしたり、電子文書の送信時間を短縮したりしている。

【0003】なお、以下の説明では、情報理論で用いられる呼称を踏襲して、データの1ワード単位を「文字」と呼び、文字が任意数つながったものを「文字列」と呼ぶ。また、データを文字から成るものとみなして、任意の種類のデータを圧縮することの可能な、いわゆる、ユニバーサル符号化の技術は、広く研究され普及している。

【0004】一方、電子文書においては、タグを付けて構造化した文書が広く用いられている。このような構造化文書を用いる場合、「文字データ」と「マーク付け」(markup) とに分けて処理が行なわれる。ここで、「マーク」とは、タグその他の構造化情報の総称であり、具体的には、開始タグ、終了タグ、空要素タグ、実体参照、文字参照、注釈、CDATAセクションの区切り子、文書型宣言、処理命令などを指す。

【0005】構造化文書の代表例としては、大規模保存データベース向けのSGML (Standard Generalized Markup Language) や、WWW (World Wide Web) 向けに簡便な構成をもつHTML (Hyper Text Markup Language) や、SGMLをmarkup向けに簡略化したXML (eXtensible Markup Language) などがある。

【0006】SGMLは、官公庁、企業等における大規

模保存文書データベースのビューアへの適用、データベース検索、製品の開発作業の同時並行化、出版(CAD、電子本、データベース)のために利用されるほか、データ交換のための中間言語としても利用される。また、HTMLはWWWとともに世界的に普及している。XMLは、HTMLを補うものとして、最近、特に注目を浴びている。このXMLは、インターネット上で文書を取り扱うためだけではなく、携帯電話、カーナビゲーション等のあらゆる情報機器が通信するための媒介として利用されつつある。

【0007】XML文書は、大きく分けて、XML宣言と、文書型定義(Document Type Definition; DTD)と、文書実現値(XMLインスタンス)との3つの部分(構成要素)から成る。また、処理上の観点から見ると、XML文書は、整形式(well-formed)と検証済み(valid)との2つに分類される。

【0008】ここで、XML宣言は、SGML宣言とは全く異なるもので、単にXMLのバージョン宣言、文字コードの宣言などを行なう簡単なものであり、DTD

は、SGMLと同じく、タグ付き文書に現れる要素、属性、エンティティの定義を行なう部分であり、XMLインスタンスは、実際のタグ付き文書が書かれる部分である。

【0009】また、整形式XML文書とは、開始タグと終了タグとの対応がとれており、XMLで規定したタグ付け規則に従ってXMLインスタンスが書かれたものであり、検証済みXML文書とは、DTDの中の要素型宣言、属性の型などに従ってタグ付けが行なわれたものである。この検証済みXML文書は、当然、整形式XML文書としてのタグ付け規則に従っていることが前提条件である。

【0010】上記構成要素と上述のごとく分類されたXML文書との対応関係、並びに、上記構成要素とSGML文書やHTML文書との対応関係(必須か否か)を、表1に示す。

【0011】

【表1】

各種宣言	文書型定義	文書実現値
整形式XML文書	△(XML宣言)	△○
検証済みXML文書	○(XML宣言)	○○
SGML文書	○(SGML宣言)	○○
HTML文書	△(HTML宣言)	○○

○: 必須 △: 必須ではない

【0012】この表1に示すように、整形式XML文書では、文書実現値のみが必須の構成要素であり、検証済みXML文書では、文書型定義および文書実現値が必須の構成要素である。また、SGML文書では、全ての構成要素が必須であり、HTML文書では、HTML宣言以外は必須の構成要素となっている。

【0013】XMLでは、文書を、階層構造をもった要素の集合としてとらえ、各要素を識別するために使用されるマークがタグである。文書の要素が始まったことを示すタグは開始タグと呼ばれ、その要素が終わつたことを示すタグは終了タグと呼ばれ、これら2つのタグで挟まれた部分が要素の内容となる。

【0014】要素に対する基本的なタグ付けは、図38(A)に示す通りである。この図38(A)に示す例は、1つの要素を表す際のタグ付けであるが、XMLインスタンスには、実際にはいくつの要素が存在し、それらの要素が階層構造になっている。つまり、ある要素の下に別の要素群が存在することがあり、このような階層構造を表現するためには、図38(B)に示すように、タグを入れ子にする。図38(B)に示すように、

要素aの下位にくる要素bと要素cとは、階層構造において同列に位置するもので、「」という親要素に対し兄弟関係にある。このような兄弟関係にある要素は、兄の要素bの終了タグのすぐあとに弟の要素cの開始タグを書くことになる。また、要素の内容として、平文(テキスト)と混在させる形で下位の要素を書くこともできる。その場合、図38(C)に示すようなタグ付けを行なう。

【0015】さらに、タグは、要素の構造を表現するだけでなく、要素になんらかの付属情報(属性)を与えることもできる。つまり、「要素のタイプを区別したい」、「要素に一意な識別子を付けて別の所から参照したい」などの理由から、付属情報としての属性を要素に与える場合がある。この属性は、図39に示すように、属性名と属性値との対で表され、開始タグの中に書き込まれる。XML文書の処理上の区分は、さらに、構成の違いにより、表2のように5通り(パターン①~⑤)に分類される。

【0016】

【表2】

構成上の区分	構成の型	文書型定義	特徴
整形式 XML文書	①DTDなし	なし	DTDなしでも解釈可能。
	②実体宣言を含むDTD	あり	実体参照を用いて内容中の長い文字列を短い文字列で置き換えるべく、DTDで対応関係を定義。
	③外部への実体宣言を含むDTD	あり	外部のファイルを内容中で引用するために実体参照を利用。DTDで対応関係を定義。
検証済み XML文書	④内部にDTD記述 (SGMLと共通)	あり	XML文書中のDTDで、要素型宣言、属性型宣言を定義。
	⑤外部からのDTD利用	あり	外部ファイルのDTDで、要素型宣言、属性型宣言を定義。

【0017】この表2に示すごとく、XML文書は、DTDをもたないもの(パターン①)と、実体宣言(エンティティ宣言)を含むDTDをもつもの(パターン②)と、外部への実体宣言を含むDTDをもつもの(パターン③)と、内部にDTDを記述したもの(パターン④: SGMLと共通)と、外部ファイルのDTDを利用するもの(パターン⑤: SGML、HTMLと共通)との5通りに分類される。

【0018】パターン①～④に対応するXML文書は、「整形式XML文書」と呼ばれ、DTDによる検証を要することなくタグを設定することができる。また、パターン④、⑤に対応するXML文書は、「検証済みXML文書」と呼ばれる。パターン①～④に対応するXMLやSGMLでは、利用者がタグを自由に設定することができる。パターン⑤では、XMLもSGMLも文書内の前のDTDでタグを定義することができる。一方、パターン⑤だけに対応しているHTMLは、外部ファイルのDTD(W3C(World Wide Web Consortium)発行)のみに依存し、利用者が自由にタグを設定することはできない。

【0019】パターン①は、XMLインスタンスのみを有し、検証済みXML文書としてのチェックを行なわないで、DTDを完全に取り払った、最もシンプルな整形式XML文書であり、DTDがなくてもXMLインスタンスの内容を解釈可能なものである。

【0020】パターン②は、置換文字列定義(実体宣言)を含むDTDを有し、XMLインスタンス内で短縮文字列を使用するために、DTDにおける実体宣言でそれらの短縮文字列を宣言した整形式XML文書である。このパターン②では、実体参照を用いて、XMLインスタンスの内容中の長い文字列を短い文字列と置き換えるべく、DTDにおいて長い文字列と長い文字列との対応関係が定義される。

【0021】パターン③は、複数のファイルでXML文書を作成するために、DTDにおける実体宣言で、それらのファイルを宣言した整形式XML文書である。このパターン③では、外部のファイルをXMLインスタンスの内容中で引用するために実体参照を用いており、DTDにおいて、XMLインスタンス内で用いられる短い短

い文字列と、実際のファイルを指定する情報との対応関係が定義される。

【0022】パターン④は、XMLインスタンスに添付されたDTD(DOCTYPE宣言)において、検証済みXML文書としてのチェックに必要な要素型宣言および属性リスト宣言を定義するものである。パターン④は、外部ファイルに存在するDTDで要素型宣言および属性リスト宣言を定義するものであり、XMLインスタンスに添付されたDTD(DOCTYPE宣言)において、その外部ファイルを指定する情報が記述されている。

【0023】あるXML文書が、整形式XML文書として解釈されるか、検証済みXML文書として解釈されるかは、XML文書を解釈するソフトウェアであるXMLプロセッサ(XMLリバーサー)に依存する。このXMLプロセッサは、図40に示すごとく、XML文書を解析し、整形式XML文書としてのチェックおよび検証済みXML文書としてのチェックを行なってから、チェックを終えた(本構造として表された)XML文書を、ブラウザなどの他の応用ソフトウェアに渡す機能を果たすものである。

【0024】そして、上述したような構造化文書を圧縮する際には、前述したように、その構造化文書を文字データの集まりとみなして圧縮を行なうユニバーサル符号化の技術が利用されている。

【0025】従来の構造化文書の圧縮手法としては、大別して下記2通りの手法(a)、(b)がある。

(a) タグを元の位置から移動せず、タグによって挟まれた平文の部分のみを圧縮する手法。

(b) タグのみを文書実現體(インスタンス)の先頭に移動することにより、タグどうし平文どうしをそれぞれまとめて圧縮する手法。

【0026】手法(a)では、タグそのものを圧縮しない。通常、タグだけで構造化文書の30%前後の容量を占めるため、タグを圧縮しなければ構造化文書の圧縮率が低下することになる。手法(b)では、圧縮された文書を伸長して復元する際にタグを元の位置に戻すべく、タグの元の位置に2バイトの識別符号を付しておかなければならず、それだけ圧縮率が低下することになる。

【0027】また、例えばHTML文書をユニバーサル

符号化技術により圧縮する場合、HTML文書の木構造で所定の深さ以下の内容を圧縮する手法や、HTML文書のタグ表現の冗長部分を検出してより簡潔な表現に置換する手法などもあるが、前者の手法では、木構造で所定の深さ以下の文書構造は、伸長・復元しない限り分からぬほか、後者の手法では、圧縮対象文書がHTML文書であるため、要素名や属性名などを圧縮することができない。

【0028】

【発明が解決しようとする課題】上述した通り、構造化文書においてタグは検索には必要であるが、上記手法(a)のようにタグだけが元のまま保存されると、構造化文書の圧縮率が悪化するし、上記手法(b)のようにタグだけ圧縮すると、圧縮ファイルの検索機能が失われ、圧縮状態において検索することができなくなる。

【0029】通常、圧縮したファイルは200KB/s程度で伸長することができるため、例えば日本工業規格A4判サイズで1頁分のデータが4~6KB程度であるSGML/XML文書は、0.02~0.03秒で伸長され閲覧可能になる。一方、データベース検索方式では0.08秒程度の処理時間で検索を行なっている。

【0030】構造化文書のデータベースに圧縮方式を適用した場合、圧縮する単位の大きさにもよるが、伸長してからの検索を行なうとなると、検索までの時間は0.1秒を超えることもある。このため、検索までの時間が0.1秒を超えてよいか否かで圧縮方式の選択が変わってくる。上述のように、タグの圧縮を行なわないと、構造化文書の圧縮率が低くなり、文書データの格納効率が低下するので、大規模なデータベースを取り扱うシステムでは好ましくない。

【0031】一方、XMLで記述された部品表や価格表等では、短い語句(内容)を挟んだ開始タグと終了タグとの対のような冗長な表現(図2(A)および図2(B)参照)が頻繁に現われるが、このような場合に、検索可能な状態を保持したままタグにかかる部分を圧縮できるようにすることが望まれている。

【0032】実際のXML文書等のデータにおいてタグで挟まれた「内容」のデータ長は短い場合が多い。具体的には、20バイト、日本語で10文字程度である。通常、短いデータは圧縮し難い。しかも、検索のキーワードとして「内容」の一部を残す場合、その「内容」のデータが短いと、「内容」を圧縮することなくそのまま残すことになり、結局、構造化文書の圧縮率が低下してしまう。

【0033】本発明は、このような課題に鑑み創案されたもので、構造化文書の特徴を損なうことなくタグ部分の圧縮を可能にし、構造化文書の圧縮率の向上をはかる、構造化文書の圧縮方法および圧縮装置並びに構造化文書圧縮プログラムを記録したコンピュータ読取可能な記録媒体を提供することを目的とする。

【0034】

【課題を解決するための手段】上記目的を達成するため、本発明の構造化文書の圧縮方法(請求項1)は、構造化文書を成す文書実現値における要素の木構造を解析する文書実現値解析ステップと、該文書実現値解析ステップでの解析結果に従い、該木構造の葉となる要素(以下、葉要素という)についての情報を、該葉要素の親要素の属性として該親要素の開始タグ内に移す文書実現値構成変更ステップとを有することを特徴としている。

【0035】上記文書実現値構成変更ステップにおいて、該葉要素についての開始タグ、終了タグおよび内容を該文書実現値から削除し、該葉要素についての情報である要素名および内容を、それぞれ該親要素の属性名および属性値として該親要素の開始タグ内に付加してもよい(請求項2)、このとき、該葉要素の開始タグ内に該葉要素についての情報である属性が記述されている場合、該属性にかかる属性名および属性値を、それぞれ該親要素の属性名および属性値として該親要素の開始タグ内に付加してもよい(請求項3)。また、上記文書実現値構成変更ステップにおいて、該親要素の終了タグを削除するとともに、該葉要素の開始タグを空要素タグに変更してもよい(請求項4)。

【0036】さらに、該構造化文書を成す文書型定義における要素の木構造を解析する文書型定義解析ステップと、該文書型定義解析ステップでの解析結果に従い、該木構造の葉となる要素(以下、葉要素という)についての情報を、該文書型定義から削除し、該葉要素の属性として該文書型定義で再定義する文書型定義構成変更ステップとをさらにそなえてよい(請求項5)。このとき、該文書型定義構成変更ステップにおいて、該葉要素の要素型宣言を該文書型定義から削除するとともに該葉要素にかかる記述を該葉要素の要素型宣言から削除し、該葉要素の要素型宣言にかかる情報を、該親要素の属性として該親要素の属性リスト宣言で再定義してもよい(請求項6)、さらに、該文書型定義で該葉要素の属性が該葉要素の属性リスト宣言により定義されている場合、該葉要素の属性リスト宣言を該文書型定義から削除し、該葉要素の属性を、該葉要素の属性として該親要素の属性リスト宣言で再定義してもよい(請求項7)。

【0037】本発明の構造化文書圧縮方法(請求項8)は、構造化文書を成す文書実現値のタグ内の記述を解析する文書実現値解析ステップと、該文書実現値解析ステップでの解析結果に従って、該文書実現値のタグ内に記述された文字列と該文字列よりも短く且つ該文字列を特定しうる短縮文字列とを対応させるタグ辞書を作成するタグ辞書作成ステップと、該タグ辞書作成ステップで作成された該タグ辞書を用いて、該文書実現値のタグ内に記述された文字列と、当該文字列に対応する短縮文字列に置き換える文書実現値文字列置換ステップとを有することを特徴としている。

【0038】本発明の構造化文書圧縮方法（請求項9）は、構造化文書を成す文書実現値のタグ内の記述を解析する文書実現値解析ステップと、該構造化文書を成す文書型定義の記述を解析する文書型定義解析ステップと、該文書実現値解析ステップおよび該文書型定義解析ステップでの解析結果に従って、該文書実現値のタグ内および該文書型定義に記述された文字列と該文字列よりも短く且つ該文字列を特定しうる短縮文字列とを対応させるタグ辞書を作成するタグ辞書作成ステップと作成された該タグ辞書を用いて、該文書実現値のタグ内に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書実現値文字列置換ステップと、該タグ辞書作成ステップで作成された該タグ辞書を用いて、該文書実現値に記述された文字列を、当該文字列よりも短く且つ該文字列を特定しうる短縮文字列と対応させるタグ辞書を作成するタグ辞書を用いて、該文書型定義に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書型定義文字列置換ステップとを有することを特徴としている。

【0039】このとき、該タグ内もしくは該文書型定義に記述された要素名および属性名を前記文字列として扱い、該要素名および該属性名を該短縮文字列に置き換えるてもよい（請求項10）。また、単語文字列と該単語文字列よりも短く且つ該単語文字列を特定しうる短縮文字列とを対応させる単語辞書を用いて、該文書実現値の内容に含まれる単語文字列を、当該単語文字列に対応する短縮文字列に置き換える単語文字列置換ステップをそなえてよい（請求項11）、さらに、該タグ内もしくは該文書型定義に記述された文字列を該短縮文字列に置き換えるとともに該単語文字列を該短縮文字列に置き換えた後に、これらの文字列を可変長符号化により圧縮する可変長符号化ステップをそなえてよい（請求項2）。

【0040】一方、本発明の構造化文書圧縮装置（請求項13）は、構造化文書を成す文書実現値における要素の木構造を解析する文書実現値解析部と、該文書実現値解析部による解析結果に従い、該木構造の葉となる要素（以下、葉要素という）についての情報を、該葉要素の親要素の属性として該親要素の開始タグ内に移す文書実現値構成変更部とをそなえて構成されたことを特徴としている。

【0041】このとき、該文書実現値構成部が、該葉要素についての開始タグ、終了タグおよび内容を該文書実現値から削除し、該葉要素についての情報である要素名および内容を、それぞれ該親要素の属性名および属性値として該親要素の開始タグ内に付加してもよく（請求項14）、さらに、該葉要素の開始タグ内に該葉要素についての情報である属性が記述されている場合、該属性にかかる属性名および属性値を、それぞれ該親要素の属性名および属性値として該親要素の開始タグ内に付加してもよいし（請求項15）、該親要素の終了タグを削除するとともに、該親要素の開始タグを空要素タグに変更し

てもよい（請求項16）。

【0042】また、該構造化文書を成す文書型定義における要素の木構造を解析する文書型定義解析部と、該文書型定義解析部による解析結果に従い、該木構造の葉となる要素（以下、葉要素という）についての情報を、該文書型定義から削除し、該葉要素の親要素の属性として該文書型定義で再定義する文書型定義構成変更部とをそなえてよい（請求項17）。

【0043】本発明の構造化文書圧縮装置（請求項18）は、構造化文書を成す文書実現値のタグ内の記述を解析する文書実現値解析部と、該文書実現値解析部による解析結果に従って、該文書実現値のタグ内に記述された文字列と該文字列よりも短く且つ該文字列を特定しうる短縮文字列とを対応させるタグ辞書を作成するタグ辞書作成部と、該タグ辞書作成部により作成された該タグ辞書を用いて、該文書実現値のタグ内に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書実現値文字列置換部をそなえて構成されたことを特徴としている。

【0044】本発明の構造化文書圧縮装置（請求項19）は、構造化文書を成す文書実現値のタグ内の記述を解析する文書実現値解析部と、該構造化文書を成す文書型定義の記述を解析する文書型定義解析部と、該文書実現値解析部および該文書型定義解析部による解析結果に従って、該文書実現値のタグ内および該文書型定義に記述された文字列と該文字列よりも短く且つ該文字列を特定しうる短縮文字列とを対応させるタグ辞書を作成するタグ辞書作成部と、該タグ辞書作成部により作成された該タグ辞書を用いて、該文書実現値のタグ内に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書実現値文字列置換部をそなえて構成されたことを特徴としている。

【0045】さらに、本発明の記録媒体（請求項21）は、構造化文書を圧縮する機能をコンピュータにより実現するための構造化文書圧縮プログラムを格納したコンピュータ読取可能なものであって、該構造化文書圧縮プログラムが、該構造化文書を成す文書実現値における要素の木構造を解析する文書実現値解析部、および、該文書実現値解析部による解析結果に従い、該木構造の葉となる要素（以下、葉要素という）についての情報を、該葉要素の親要素の属性として該親要素の開始タグ内に移す文書実現値構成変更部として、該コンピュータを機能させることを特徴としている。

【0046】このとき、該構造化文書圧縮プログラム

が、該文書実現値構成部により、該要素についての開始タグ、終了タグおよび内容を該文書実現値から削除し、該要素についての情報である要素名および内容を、それぞれ該要素の属性名および属性値として該要素の開始タグ内に付加するよう、該コンピュータを機能させてもよいし（請求項22）、さらに、該要素の開始タグ内に該要素についての情報である属性が記述されている場合、該文書実現値構成変更部により、該属性にかかる属性名および属性値を、それを該要素の属性名および属性値として該要素の開始タグ内に付加するよう、該コンピュータを機能させてもよいし（請求項23）、該文書実現値構成変更部により、該要素の終了タグを削除するとともに該要素の開始タグを空要素タグに変更するよう、該コンピュータを機能させてもよい（請求項24）。

【0047】また、該構造化文書圧縮プログラムが、該構造化文書を成す文書型定義における要素の木構造を解析する文書型定義解析部、および、該文書型定義解析部による解析結果に従い、該木構造の葉となる要素（以下、葉要素という）についての情報を、該文書型定義から削除し、該要素の親要素の属性として該文書型定義で再定義する文書型定義構成変更部として、該コンピュータを機能させてもよい（請求項25）。

【0048】本発明の記録媒体（請求項26）は、構造化文書を圧縮する機能をコンピュータにより実現するための構造化文書圧縮プログラムを格納したコンピュータ読取可能なものであって、該構造化文書圧縮プログラムが、該構造化文書を成す文書実現値のタグ内の記述を解析する文書実現値解析部、該文書実現値解析部による解析結果に従って、該文書実現値のタグ内に記述された文字列と該文字列よりも短く且つ該文字列を特定する短縮文字列とを対応させるタグ辞書を作成するタグ辞書作成部、および、該タグ辞書作成部により作成された該タグ辞書を用いて、該文書実現値のタグ内に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書実現値文字列置換部として、該コンピュータを機能させることを特徴としている。

【0049】本発明の記録媒体（請求項27）は、構造化文書を圧縮する機能をコンピュータにより実現するための構造化文書圧縮プログラムを格納したコンピュータ読取可能なものであって、該構造化文書圧縮プログラムが、該構造化文書を成す文書実現値のタグ内の記述を解析する文書実現値解析部、該構造化文書を成す文書型定義の記述を解析する文書型定義解析部、該文書実現値解析部および該文書型定義解析部による解析結果に従って、該文書実現値のタグ内および該文書型定義に記述された文字列と該文字列よりも短く且つ該文字列を特定する短縮文字列とを対応させるタグ辞書を作成するタグ辞書作成部、該タグ辞書作成部により作成された該タグ辞書を用いて、該文書実現値のタグ内に記述された文字

列を、当該文字列に対応する短縮文字列に置き換える文書実現値文字列置換部、および、該タグ辞書作成部により作成された該タグ辞書を用いて、該文書型定義に記述された文字列を、当該文字列に対応する短縮文字列に置き換える文書型定義文字列置換部として、該コンピュータを機能させることを特徴としている。このとき、該構造化文書圧縮プログラムが、該タグ内もしくは該文書型定義に記述された要素名および属性名を前記文字列として扱い、該コンピュータに、該要素名および該属性名を該短縮文字列に置換させてもよい（請求項28）。

【0050】

【発明の実施の形態】以下、図面を参照して本発明の実施の形態を説明する。

〔1〕第1実施形態の説明

まず、図2～図4を参照しながら、本発明の第1実施形態における構造化文書の圧縮原理について説明する。なお、図2（A）～図2（D）および図3（A）～図3（C）はいずれも第1実施形態における構造化文書（文書実現値）の圧縮原理を説明するための図、図4（A）および図4（B）はいずれも第1実施形態における構造化文書（DTD）の圧縮原理を説明するための図である。なお、以下、本発明の第1実施形態では、構造化文書がXML文書である場合について説明する。

【0051】前述した通り、XMLで記述された部品表や価格表等では、図2（A）および図2（B）に示すとく、短い語句（平文）を内容として扱んだ開始タグと終了タグとの対のような冗長な表現が頻繁に現われる。ここで、図2（A）には、ある親要素（要素1）の下に2つの子要素（子要素をもたない要素2および要素3）が存在する場合の、XMLの一般的な記述例が示されている。そして、図2（B）には、図2（A）に示した一般的な記述例に対応した具体的な記述例が示されている。なお、以下、子要素をもたない要素を葉の要素、葉要素もしくは単に葉と呼ぶ場合がある。

【0052】図2（A）に示す一般的な記述例において、要素1は、要素名1を指定されるとともに属性情報（属性名1および属性値1）を指定され、要素1の子要素である要素2は、要素名2を指定されて内容2を有し、要素2と同じく要素1の子要素（要素2と兄弟関係）である要素3は、要素名3を指定されるとともに属性情報（属性名3および属性値3）を指定されて内容3を有している。

【0053】そして、図2（B）に示す具体的な記述例では、要素名1が「book」、属性名1が「field」、属性値1が「本」、要素名2が「title」、内容2が「XML入門」、要素名3が「author」、属性名3が「yea」、属性値3が「1955」、内容3が「佐藤元」となっている。これらの図2（A）および図2（B）に示す記述例の木構造を図2（C）に示す。また、これらの図2（A）～図2（C）に示す例についての、葉の要素一覧

表を図2 (D) に示す。

【0054】なお、図2 (B) に示す記述例において、1行目の記述<book field="本">が要素「book」の開始タグで、5行目の記述</book>が要素「book」の終了タグであり、これらのタグにより括られた、2~4行目の記述が要素「book」の内容を示している。1行目の開始タグ内の記述「field="本"」は、要素「book」の属性情報（属性名が「field」で属性値が「本」）を示している。

【0055】また、2行目において、記述<title>が要素「title」の開始タグで、記述</title>が要素「title」の終了タグであり、これらのタグ間の記述「XML入門」が要素「title」の内容である。同様に、3行目の記述<author year="1955">が要素「author」の開始タグで、4行目の記述</author>が要素「author」の終了タグであり、これらのタグ間の記述「佐藤元」が要素「author」の内容である。3行目の開始タグ内の記述「year="1955"」は、要素「author」の属性情報（属性名が「year」で属性値が「1955」）を示している。

【0056】本発明の第1実施形態では、図2 (A) ~図2 (D) に示すごとく、子要素をもたず木構造の葉として並んでいる要素（以下、葉の要素、葉要素もしくは単に葉と呼ぶ場合がある）を検出し、図2 (D) に示すような葉の要素一覧表についてのファイルを出力する。そして、そのファイルに基づいて、図3 (A) および図3 (B) に示すように、葉の要素名や内容を上位の要素1（親要素）の属性に置き換えてその葉の要素を削除するとともに、要素1の開始タグを空要素タグに変更する。このとき、要素3の属性情報（属性名3および属性値3）も、要素3の要素名3や内容3と対等な、要素1の属性として並ぶ。

【0057】ここで、図3 (A) や図3 (B) に示すごとく、要素名や属性情報を< >と併んで記述されたタグは、内容をもたない空要素タグである。このとき、属性情報については必ずしも記述・指定しなくてよい。図3 (A) には、図2 (A) に示した一般的な記述例を第1実施形態の圧縮方法により圧縮した結果得られる記述が示され、図3 (B) には、図2 (B) に示した具体的な記述例を第1実施形態の圧縮方法により圧縮した結果得られる記述が示されている。

【0058】これらの図3 (A) および図3 (B) に示すように、要素1の開始タグ（空要素タグ）においては、要素名1および属性情報（属性名1および属性値1）が指定されるだけでなく、要素名2および内容2がそれぞれ要素1の第2の属性名および属性値として指定され、要素名3および内容3がそれぞれ要素1の第3の属性名および属性値として指定され、要素名3および属性値3がそれぞれ要素1の第4の属性名および属性値として指定されている。なお、図3 (C) は、図3 (A)

および図3 (B) に示した開始タグ（空要素タグ）をもつ要素1の構造を図式表現したものである。

【0059】一方、図2や図3を参照しながら上述したことなく圧縮処理を行なった構造化文書がDTDを有している場合には、その圧縮処理に対応して、図4 (A) および図4 (B) に示すごとくDTDの変更（圧縮）も行なわれる。即ち、図4 (A) には、図2 (B) に示すXML文書を定義する、変更前（圧縮前）のDTDが示されている。この図4 (A) に示すDTDにおいて、1行目の記述は文書型宣言（DOCTYPE宣言）であり、ここでは、この文書の文書型名つまり最上位要素の要素名が「book」であることが宣言されている。

【0060】そして、1行目末尾の【" 】と7行目の【>】との間ににおける記述（2~6行目の記述）が、この文書の構造を定義している。2行目の要素型宣言の要素名と文書型名は一致する必要があり、ここでは、その規則に従い最上位要素の要素名として「book」が指定されている。また、この親要素「book」の下に要素名「title」および「author」の2つの子要素が並んで存在することが、2行目の要素型宣言内において、内容モデル記述「(title,author)」により宣言されている。つまり、要素「book」は2つの子要素「title」および「author」から構成されることが宣言されている。

【0061】さらに、3行目の要素型宣言により、要素の要素名として「title」が指定されるとともに、この要素「title」の内容が文字データ（#PCDATA）であることが宣言されている。同様に、4行目の要素型宣言により、要素の要素名として「author」が指定されるとともに、この要素「author」の内容が文字データ（#PCDATA）であることが宣言されている。これらの要素型宣言では、各要素の下における子要素の存在は宣言されていない。つまり、これらの要素を親とする子要素は存在しておらず、これらの要素は、木構造の葉を成す要素である。

【0062】また、5行目の属性リスト宣言では、要素「book」に伴う属性として、属性名「field」と、属性値についての3種類の候補「本」、「雑誌」、「小冊子」と、デフォルト値「本」とが宣言されている。さらに、6行目の属性リスト宣言では、要素「author」に伴う属性として、著者の生年を示す属性名「year」と、その属性値のデータ型（CDATA）とが宣言されている。なお、7行目の記述「】」は、これで1行目の文書型宣言の内部サブセット記述部分が終了することを示している。

【0063】そして、図4 (B) には、図3 (B) に示すXML文書を定義する、変更後（圧縮後）のDTDが示されており、この図4 (B) に示すDTDにおいては、図4 (A) に示すDTDに存在していた、要素「title」および要素「author」についての要素型宣言が消えるとともに、2行目の親要素「book」の要素型宣言内

における子要素の定義（内容モデル）が消えている。また、要素「author」に伴う属性「year」も消えている。

【0064】代わりに、3～6行目の属性リスト宣言において、親要素「book」の属性として、新たに属性名「title」、「author」、「year」が付加されている。そして、これらの属性名「title」、「author」の属性値の候補として「#PCDATA」が宣言されるとともに、属性名「year」の属性値の候補として「CDATA」が宣言されている。

【0065】ここで、「#PCDATA」としては、文字データのほかに、タグや、実体参照と呼ばれるマークが含まれる。なお、実体参照は、構造化文書を記述する際、文字データ中で特定の文字列を所定のマークで表現し、文字データを応用ソフトウェアで渡す際には、マークの部分に元の文字列を代入する機能である。ただし、属性値の中で実体参照を行なう場合、その属性値にかかる文書と同じ文書内の文字列を参照することは許されているが、外部ファイルにおける文字列を参照することは許されていない。また、生年を表す属性値としては、純粋な文字データであるCDATAが宣言されている。

【0066】第1実施形態の構造化文書の圧縮手法では、上述のごとく文書実現値の圧縮処理に合わせてDTDも変更・圧縮するため、図4 0で前述したXMLプロセッサは、上述の圧縮に呼応した形で、図3 (A) や図3 (B) に示すような新しいXML文書を正しく検討することができる。

【0067】また、上述のごとく、従来、「要素」として表現された情報が親要素の「属性」に変換されるとともに、その変換に伴うXML文書の構造変更に合わせてDTDも変更される。これにより、従来、「要素」として表現された情報は、親要素の「属性」として検出されるようになる。従って、「要素名のバイト数」+「3バイト」分だけ、XML文書の表現が圧縮・節約されることがある。

【0068】以下、図1～図13を参照しながら、本発明の第1実施形態について、より詳細かつ具体的に説明する。図1は本発明の第1実施形態としての構造化文書の圧縮装置の機能構成を示すブロック図であり、この図1に示すように、第1実施形態の圧縮装置は、文書記憶部10、文書実現値解析部20、DTD解析部30、文書実現値構成変更部40、DTD構成変更部50、新規DTDファイル作成部60および新旧DTD対応表示出力部70を有して構成されている。

【0069】ここで、本実施形態の圧縮装置は、CPU、RAM、ROMなどをバスラインにより接続して構成される。パソコン等のコンピュータシステムにより実現されるものである。つまり、RAMやROMが文書記憶部10としての機能を果たすほか、RAMには、文書実現値解析部20、DTD解析部30、文書実現値構成変更部40、DTD構成変更部50、新規DTDファイ

ル作成部60および新旧DTD対応表示出力部70を実現するためのアプリケーションプログラムが格納されている。

【0070】そして、CPUが、上記アプリケーションプログラムを実行することにより、文書実現値解析部20、DTD解析部30、文書実現値構成変更部40、DTD構成変更部50、新規DTDファイル作成部60および新旧DTD対応表示出力部70としての機能（その詳細については後述）が実現され、第1実施形態の構造化文書の圧縮装置が実現されるようになっている。

【0071】この第1実施形態の圧縮装置を実現するためのプログラムは、例えばフレキシブルディスク、CD-ROM等の、コンピュータ読取可能な記録媒体に記録された形態で提供される。そして、コンピュータはその記録媒体からプログラムを読み取って内部記憶装置または外部記憶装置に転送し格納して用いる。また、そのプログラムを、例えば磁気ディスク、光ディスク、光磁気ディスク等の記憶装置（記録媒体）に記録しておき、その記憶装置から通信経路を介してコンピュータに提供してもよい。

【0072】そして、第1実施形態の圧縮装置としての機能をコンピュータにより実現する際には、内部記憶装置（例えばRAM）に格納された上記プログラムがコンピュータのマイクロプロセッサ（例えばCPU）によって実行される。このとき、記録媒体に記録されたプログラムをマイクロプロセッサが直接読み取って実行してもよい。

【0073】なお、本実施形態において、コンピュータとは、ハードウェアとオペレーティングシステムとを含む概念であり、オペレーティングシステムの制御の下で動作するハードウェアを意味している。また、オペレーティングシステムが不要でアプリケーションプログラム単独でハードウェアを動作させるような場合には、そのハードウェア自体がコンピュータに相当する。ハードウェアは、少なくとも、CPU等のマイクロプロセッサと、記録媒体に記録されたコンピュータプログラムを読み取るための手段とをそなえている。

【0074】上記アプリケーションプログラムは、このようなコンピュータに、文書実現値解析部20、DTD解析部30、文書実現値構成変更部40、DTD構成変更部50、新規DTDファイル作成部60および新旧DTD対応表示出力部70としての機能を実現させるプログラムコードを含んでいる。また、その機能の一部は、アプリケーションプログラムではなくオペレーティングシステムによって実現されてもよい。

【0075】さらに、本実施形態における記録媒体としては、上述したフレキシブルディスク、CD-ROM、磁気ディスク、光ディスク、光磁気ディスクのほか、ICカード、ROMカードリッジ、磁気テープ、バンチカード、コンピュータの内部記憶装置（RAMやROMな

どのメモリ)、外部記憶装置等や、バーコードなどの符号が印刷された印刷物等の、コンピュータ読取可能な種々の媒体を利用することができる。

【0076】さて、図1に示す第1実施形態の圧縮装置において、文書記憶部10は、構造化文書であるXML文書を記憶するもので、本実施形態では、圧縮前および圧縮後のいずれのXML文書も記憶するほか、後述する新旧DTD対応表も記憶するものである。

【0077】文書実現値解析部20は、XML文書を成す文書実現値を解析し、文書実現値における要素の木構造(親子関係)や、文書実現値の記述を解析するもので、その解析手順については、図6に示すフローチャートを参照しながら後述する。そして、本実施形態の文書実現値解析部20は、要素の木構造(親子関係)の解析結果として、例えば図2(D)に示すような葉の要素一覧表(ファイル)を出力する。

【0078】DTD解析部(文書型定義解析部)30は、XML文書が検証済みXML文書である場合(つまりパターン①または⑤のXML文書である場合)に、そのDTDを解析し、DTDにおける要素の木構造(親子関係)や、DTDの記述を解析するもので、その解析手順については図7に示すフローチャートを参照しながら後述する。そして、本実施形態のDTD解析部30も、要素の木構造(親子関係)の解析結果として、例えば図2(D)に示すような葉の要素一覧表(ファイル)を出力する。ただし、XML文書がパターン④である場合、DTDは文書記憶部10から読み込まれるが、XML文書がパターン⑥である場合、DTDは外部ファイル10から読み込まれる。

【0079】このとき、文書実現値解析部20やDTD解析部30から出力される葉の要素の一覧表は、前述した通り、子要素をもたず木構造の葉として並んでいる要素を検出し、その葉の要素と親要素との対応関係を明記したものである。文書実現値解析部20は、文書実現値の表現を簡潔にすべく、文書実現値解析部20による解析結果(葉の要素一覧表)に従って、文書実現値における葉要素についての情報を、その葉要素の親要素の属性名として親要素の属性リスト内に移動させるもので、以下のよう構成変更処理(a1)～(a3)を実行するものである。

【0080】(a1)葉要素についての開始タグ、終了タグおよび内容を文書実現値から削除し、葉要素についての情報である要素名および内容を、それぞれ親要素の属性名および属性値として親要素の開始タグ内に付加する。

(a2)葉要素の開始タグ内に葉要素についての情報である属性が記述されている場合、属性にかかる属性名および属性値を、それぞれ親要素の属性名および属性値として親要素の開始タグ内に付加する。

【0081】(a3)親要素の終了タグを削除するとと

もに、親要素の開始タグを空要素タグに変更する。この文書実現値構成変更部40による構成変更手順については、図8に示すフローチャートを参照しながら後述する。また、本実施形態では、文書実現値構成変更部40による構成変更結果(圧縮後の文書実現値)を、文書記憶部10に出力・格納しているが、その他の記録媒体等に出力・格納してもよい。

【0082】DTD構成変更部(文書型定義構成変更部)50は、XML文書が検証済みXML文書である場合(XML文書がパターン④または⑥である場合)に、文書実現値構成変更部40による構成変更に合わせてDTDの表現を簡潔にすべく、DTD解析部30による解析結果(葉の要素一覧表)に従って、DTDにおける葉要素についての情報を、DTDから削除し、その葉要素の親要素の属性としてDTDで再定義するもので、以下のよう構成変更処理(b1)および(b2)を実行するものである。

【0083】(b1)葉要素の要素型宣言をDTDから削除するとともに葉要素にかかる記述(内容モデルの記述)を親要素の要素型宣言から削除し、葉要素の要素型宣言にかかる情報(削除した部分にかかる情報)を、親要素の属性として親要素の属性リスト宣言で再定義する。このとき、葉要素の要素名および内容(データ型)を、それぞれ親要素の属性名および属性値として宣言・再定義する。

【0084】(b2)DTDで葉要素の属性が葉要素の属性リスト宣言により定義されている場合、葉要素の属性リスト宣言をDTDから削除し、葉要素の属性を、その葉要素の親要素の属性として親要素の属性リスト宣言で再定義する。このとき、葉要素についての属性名および属性値を、それぞれ親要素の属性名および属性値として宣言・再定義する。

【0085】このDTD構成変更部50による構成変更手順については、図9に示すフローチャートを参照しながら後述する。また、本実施形態では、XML文書がパターン④である場合、DTD構成変更部50による構成変更結果(圧縮後のDTD)を、文書実現値構成変更部40による構成変更結果(圧縮後の文書実現値)とともに文書記憶部10に出力・格納しているが、圧縮後の文書実現値とともに他の記録媒体等に出力・格納してもよい。

【0086】新規DTDファイル作成部60は、DTDが外部ファイル100に存在する場合(XML文書がパターン④である場合)、DTD構成変更部50により変更処理されたDTDについてのファイル(新規DTDファイル)を作成して外部ファイル100へ出力するものである。新旧DTD対応表出力部70は、DTDが外部ファイル100に存在する場合(XML文書がパターン④である場合)、構成変更前のDTDと構成変更後の新規DTDとの対応関係を明記した新旧DTD対応表を作

成して文書記憶部10へ出力するものである。

【0087】なお、第1実施形態の圧縮手法（要素から属性への変換）により圧縮されたXML文書は、XML文書としての特徴を全く損なわらず、圧縮状態のままで（伸長することなく）XML文書としての機能を果たすことができる、圧縮されたXML文書の伸長について特に議論する必要はない。従って、第1実施形態では、圧縮手法についてのみ説明する。

【0088】次に、図5～図13を参照しながら、第1実施形態の圧縮装置の動作について説明する。まず、図5に示すフローチャート（ステップS11～S29）に従って、第1実施形態における構造化文書（XML文書）の圧縮手順について説明する。第1実施形態の圧縮手法では、前述した通り要素から属性への変換を行なうことにより、葉の要素が消え、その葉の親要素が空要素になる。

【0089】なお、第1実施形態の圧縮装置には、図1では図示省略しているが、文書記憶部10に保存されているXML文書がパターン①～④（表2参照）のいずれのものであるからを認識するためのパターン認識機能がそなえられている。このパターン認識機能による処理は、図5に示すステップS12～S14による処理に対応している。

【0090】圧縮対象のXML文書が入力され文書記憶部10に格納されると（ステップS11）、そのXML文書に“<!DOCTYPE”が記述されているか否かを判定し（ステップS12）、記述されていない場合（ステップS12のNOルート）、そのXML文書はDTDをもたない整形式XML文書、つまりパターン①のXML文書であると認識され、後述するごくステップS15、S16およびS29が実行される。

【0091】XML文書に“<!DOCTYPE”が記述されている場合（ステップS12のYESルート）、その後に“（”が記述されているか否かを判定する（ステップS13）。“<!DOCTYPE”は記述されているが“（”が記述されていない場合（ステップS13のNOルート）、そのXML文書は、DTDを外部ファイル100として有する検証済みXML文書、つまりパターン⑤のXML文書であると認識され、後述のごくステップS21～S29が実行される。

【0092】“（”が記述されている場合（ステップS13のYESルート），“<!ELEMENT”（もしくは“<!ATTLIST”）が記述されているか否かを判定する（ステップS14）。“<!DOCTYPE”および“（”は記述されているが“<!ELEMENT”が記述されていない場合（ステップS14のNOルート）、実体宣言を含むDTDを有する整形式XML文書、つまりパターン②のXML文書であると認識され、パターン①の場合と同様、ステップS15、S16およびS29が実行される。

【0093】“<!DOCTYPE”、“（”および“<!ELEMENT”がいずれも記述されている場合（ステップS14のYESルート）、XML文書内にDTDを有する検証済みXML文書、つまりパターン④のXML文書であると認識され、ステップS17～S20およびS29が実行される。以下、各パターン①～④に対する圧縮処理について、図10～図13に示す具体例（第1例～第4例）を参照しながら説明する。

【0094】図10（A）および図10（B）はいずれも第1実施形態によるXML文書の具体的な圧縮処理（第1例）を説明するための図である。図10（A）に示す圧縮前のXML文書は、前述したパターン①のXML文書であり、1行目に、この文書がバージョン1.0のXML文書であることを示すXML宣言が記述され、2～5行目に、文書実現値が記述されている。ここに記述された文書実現値は、要素“book”的開始タグ内において属性情報（field=“本”）が記述されていない点を除けば、図2（B）に示したXML文書の記述例と同一である。

【0095】図10（A）に示すXML文書（パターン①）には“<!DOCTYPE”が記述されていないので、処理はステップS12のNOルートからステップS15へ移行し、文書実現値解析部20によって文書実現値が解析される。これにより、文書実現値において、葉となる要素がどこに記述されているかが検出され、その葉要素の要素名と、親要素の要素名との対応関係が、葉の要素一覧表として登録・出力される。図10（A）に示すXML文書の場合、図2（D）と同様の葉の要素一覧表が得られる。

【0096】そして、ステップS15で得られた、葉の要素一覧表（解析結果）に従って、文書実現値構成変更部40により、葉の要素についての要素名および内容が、それぞれ、親要素の属性名および属性値に移動・変更されるとともに、親要素の開始タグが空要素タグに変更される（ステップS16）。このとき、葉の要素“author”に付随した属性“year”は、親要素“book”的属性に代わる。

【0097】ステップS15およびS16によって、例えば図10（A）に示すXML文書は、図10（B）に示すようなXML文書に変更・圧縮されてから、圧縮文書として文書記憶部10等へ出力・格納される（ステップS29）。図10（B）に示すXML文書において、1行目のXML宣言の記述は圧縮前と変わらないが、2行目には、図10（A）における2～5行目の記述が集約されて記述されている。つまり、図10（B）に示すXML文書では、図3（B）に示した例と同様、2つの子要素“title”および“author”にかかる全ての情報が、親要素“book”的開始タグ（空要素タグ）において、親要素“book”的属性として記述される。

【0098】図11（A）および図11（B）はいずれ

も第1実施形態によるXML文書の具体的な圧縮処理(第2例)を説明するための図である。図11(A)に示す圧縮前のXML文書は、前述したパターン②のXML文書であり、1行目に、この文書がバージョン1.0のXML文書であることを示すXML宣言が記述され、2~4行目に、置換文字列定義(実体宣言)を含むDTDが記述され、5~8行目に文書実現値が記述されている。

【0099】2~4行目のDTDでは、文書型宣言に含まれる実体宣言(3行目)により、文書実現値(XMLインスタンス)内で用いられる置換文字列「XML」の実体が「Extensible Markup Language」であることが定義されている。また、5~8行目に記述された文書実現値は、図10(A)に示したXML文書の2~5行目の記述例とほぼ同一であるが、図11(A)に示す例では、6行目の要素「title」の内容として、「XML(& XML; の略称) 入門」が記述されている。

【0100】ここで、「& XML;」は、置換文字列「XML」の実体を参照することを指示する記述であり、実際に表示・印刷等によって出力される文書中では、「Extensible Markup Language」と表記されることになる。図11(A)に示すXML文書(パターン②)には、「<!DOCTYPE」および「;」がいずれも記述されているが、「<!ELEMENT」や「<!ATTLIST」が記述されていないので、処理はステップS14のNOルートからステップS15へ移行し、前述したパターン①のXML文書と同様の処理が実行される。このとき、文書実現値の内容中における実体参照の記述(図11(A)では「& XML;」)は、そのまま、親要素の属性値として取り扱われる。

【0101】これにより、例えば図11(A)に示すXML文書は、図11(B)に示すようなXML文書に変更・圧縮されてから、圧縮文書として文書記憶部10等へ出力・格納される(ステップS29)。図11(B)に示すXML文書において、1~4行目の記述は圧縮前と変わらないが、5行目には、図11(A)における5~8行目の記述が集約されて記述されている。つまり、図11(B)に示すXML文書でも、図10(B)に示した例と同様、2つの子要素「title」および「author」にかかる全ての情報が、親要素「book」の開始タグ(空要素タグ)において、親要素「book」の属性として記述される。ただし、図11(B)では、属性名「title」に対する属性値として、「XML(& XML; の略称) 入門」がそのまま記述される。

【0102】ところで、パターン③のXML文書(例えば図22(A)参照)は、前述した通り、外部ファイルを文書実現値(XMLインスタンス)の内容中で引用するために実体参照を用いるものである。第1実施形態では、文書実現値の内容を親要素の属性値として取り扱っているが、XML文書の仕様上、外部に対する実体参照

を属性値で用いることはできないため、パターン③のXML文書に、第1実施形態の圧縮手法は適用されない。

【0103】図12(A)および図12(B)はいずれも第1実施形態によるXML文書の具体的な圧縮処理(第3例)を説明するための図である。図12(A)に示す圧縮前のXML文書は、前述したパターン④のXML文書であり、1行目に、この文書がバージョン1.0のXML文書であることを示すXML宣言が記述され、2~8行目にDTDが記述され、9~12行目に文書実現値が記述されている。ここで、2~8行目に記述されたDTDは、図4(A)に示したDTDと同一であり、9~12行目に記述された文書実現値は、図2(B)に示した文書実現値の記述例と同一であるので、その説明は省略する。

【0104】図12(A)に示すXML文書(パターン④)には、「<!DOCTYPE」および「;」が記述されるとともに、「<!ELEMENT」または「<!ATTLIST」も記述されているので、処理はステップS14のYESルートからステップS17へ移行し、文書実現値解析部20によって文書実現値が解析されるとともに、DTD解析部30によってDTDが解析される(ステップS18)。これにより、文書実現値中やDTD中において、葉となる要素がどこに記述されているかが検出され、その葉要素の要素名と、親要素の要素名との対応関係、葉の要素一覧表として登録・出力される。図12(A)に示すXML文書の場合も、図2(D)と同様の葉の要素一覧表が得られる。

【0105】そして、ステップS17で得られた、葉の要素一覧表(解析結果)に従って、文書実現値構成変更部40により、葉の要素についての要素名および内容が、それぞれ、親要素の属性名および属性値に移動・変更されるとともに、親要素の開始タグが空要素タグに変更される(ステップS19)。このとき、葉の要素「author」に付随した属性「year」は、親要素「book」の属性に化れる。

【0106】また、図12(A)に示すXML文書はパターン④(即ち、その内部にDTDを記述された、検証済みXML文書)であるので、文書実現値構成変更部40による文書実現値の構成変更に合わせて、DTDの構成を、DTD構成変更部50により以下のように変更する(ステップS20)。

【0107】つまり、構成変更後の文書実現値において、親要素「book」は子要素「title」および「author」をもたなくなるので、親要素「book」についての要素型宣言内に子要素を宣言していた内容モデル(title, author)は削除される。また、構成変更後の文書実現値において、葉の要素「title」および「author」は削除され親要素「book」の属性に変更されるとともに要素「author」の属性「year」も親要素「book」の属性に変更されるので、要素「title」および「author」の要素

型宣言や要素「author」の属性リスト宣言も削除される。一方、親要素「book」は、新たに「title」、「author」および「year」を属性としてもつことになるため、親要素「book」の属性リスト宣言で、新たな属性の属性名および属性値を列挙する。

【0108】上述したステップS17～S20によって、図12 (A) に示すXML文書は、図12 (B) に示すようなXML文書に変更・圧縮されてから、圧縮文書として文書実現値10等へ出力・格納される(ステップS29)。図12 (B) に示すXML文書において、1行目および2行目の記述は圧縮前と変わらないが、3行目の要素「book」の要素型宣言からは内容モデルの記述が削除されている。また、4～7行目の記述は、図12 (A) における4～7行目の記述を「要素「book」の属性リスト宣言内にまとめたものとなっている。

【0109】さらに、図3 (B) に示した例と同様、9行目には、図12 (A) における9～12行目の記述が集約されて記述されている。つまり、図12 (B) に示すXML文書でも、図3 (B) に示した例と同様、2つの要素(要素名)「title」および「author」にかかる全ての情報が、親要素「book」の開始タグ(空要素タグ)において、親要素「book」の属性として記述される。

【0110】図13 (A)～図13 (D) はいずれも第1実施形態によるXML文書の具体的な圧縮処理(第4例)を説明するための図である。図13 (A) に示す圧縮前のXML文書は、前述したパターン⑤のXML文書であり、1行目に、この文書がバージョン1.0のXML文書であることを示すXML宣言が記述され、2行目に、外部ファイル100のDTDを指定するための情報(システム識別子)を含むDTDが記述され、3～6行目に文書実現値が記述されている。ここで、3～6行目に記述された文書実現値は、図2 (B) に示した文書実現値の記述例と同一であるので、その説明は省略する。

【0111】2行目のDTDの文書型宣言では、システム識別子“SYSTEM”により、外部ファイル100に保持されたDTD(ファイル名「.vbook.dtd」)を用いることが宣言・定義されている。そして、ファイル名「.vbook.dtd」のDTDは、図13 (A) における文書実現値の構成に対応して、図13 (B) に示すように記述されている。この図13 (B) に示すDTD(1～5行目)は、図4 (A) に示したDTDにおける2～6行目の記述例と同一であるので、その説明は省略する。

【0112】図13 (A) に示すXML文書(パターン⑤)には、“<!DOCTYPE”は記述されているが、その後には “[” が記述されることなく、外部ファイル100におけるDTDを指定するシステム識別子が記述されているので、処理はステップS13のNOルートからステップS21へ移行し、文書実現値解析部20

によって文書実現値が解析されるとともに、DTD解析部30によって、システム識別子に従って外部ファイル100から読み込まれたDTD(ファイル名「.vbook.dtd」)が解析される(ステップS22)。これにより、文書実現値中やDTD中において、葉となる要素がどこに記述されているかが検出され、その葉の要素名と、親要素の要素名との対応関係が、葉の要素一覧表として登録・出力される。図13 (A) や図13 (B) に示すXML文書の場合も、図2 (D) と同様の葉の要素一覧表が得られる。

【0113】このとき、図13 (B) に示すDTDを変更・圧縮して得られる新規のDTDのために、元のファイル名とは異なる新規のファイル名(例えば「..vbook2.dtd」)を設定して文書実現値に記入することにより、文書実現値における文書型宣言のシステム識別子“SYSTEM”により指定されるファイル名を、旧ファイル名「..vbook.dtd」から、新規ファイル名「..vbook2.dtd」に書き換える。

【0114】この後、ステップS21で得られた、葉の要素一覧表(解析結果)に従って、文書実現値構成変更部40により、葉の要素についての要素名および内容が、それぞれ、親要素の属性名および属性値に移動・変更されるとともに、親要素の開始タグが空要素タグに変更される(ステップS24)。このとき、葉の要素「author」に付随した属性「year」は、親要素「book」の属性に代わる。

【0115】これにより、図13 (A) に示すXML文書は、図13 (C) に示すようなXML文書に変更・圧縮される。図13 (C) に示すXML文書において、1行目の記述は圧縮前と変わらないが、2行目には、図13 (A) における3～6行目の記述が集約されて記述されている。つまり、図13 (C) に示すXML文書でも、図3 (B) に示した例と同様、2つの要素(要素名)「title」および「author」にかかる全ての情報が、親要素「book」の開始タグ(空要素タグ)において、親要素「book」の属性として記述される。

【0116】この後、新規DTDファイルを作成部60により、新規DTDファイルを作成し、そのDTDファイルに、外部ファイル100から読み込んだ圧縮前のDTDファイルの内容を複写してから(ステップS25)、文書実現値構成変更部40による文書実現値の構成変更に合わせ、新規ファイルにおけるDTDの構成を、DTD構成変更部50により、前述したステップS20と同様にして変更する(ステップS26)。

【0117】これにより、図13 (B) に示すDTDは、図13 (D) に示すようなDTDに変更・圧縮される。図13 (D) に示すDTDにおいて、1行目の要素「book」の要素型宣言からは内容モデルの記述が削除さ

れている。また、2~5行目の記述は、図13 (B) における2~5行目の記述を、要素「book」の属性リスト宣言内にまとめたものとなっている。

【0118】そして、DTD構成変更部5.0で変更・圧縮されたDTDのファイル（新規DTDファイル）は、新規のファイル名「..book2.dtd」を付与されて、新規DTDファイル作成部6.0から外部ファイル10.0へ出力・格納される（ステップS27）。

【0119】また、新旧DTD対応表出力部7.0によって、旧DTDと新規DTDとの対応関係（具体的には旧ファイル名と新規ファイル名との対応関係）を明記した新旧DTD対応表が作成されて文書記憶部10等へ出力・格納されるとともに（ステップS28）、ステップS24において変更・圧縮されたXML文書は、圧縮文書として文書記憶部1等へ出力・格納される（ステップS29）。その際、新旧DTD対応表は、独立したファイルではなく、圧縮文書に注釈の形で付加してもよい。

【0120】なお、第1実施形態では、圧縮したXML文書を元の状態に復元（伸長）する必要はないので、必ずしも、元のDTDの保存や新旧DTD対応表の作成を実行しなくてもよい。つまり、第1実施形態では、新規DTDファイル作成部6.0やこの新規DTDファイル作成部6.0によるステップS25、S27の処理、並びに、新旧DTD対応表出力部7.0やこの新旧DTD対応表出力部7.0によるステップS28の処理を省略することも可能である。

【0121】ただし、元のDTDや新旧DTD対応表は、第2実施形態で後述するごく圧縮されたXML文書を復元（伸長）する際に必要になるものである。第1実施形態の圧縮装置は、上述のような、元のDTDの保存機能や新旧DTD対応表の作成機能をそなえるとともに、図6を参照しながら後述するタグ辞書作成機能をそなえ、後述する第2実施形態の圧縮手法を実現することもできるように構成されている。

【0122】さて、次に、図6へ9図を参照しながら、第1実施形態の圧縮装置を構成する各部2.0、3.0、4.0および5.0の動作について説明する。まず、図6に示すフローチャート（ステップS31~S43）に従って、第1実施形態の文書実現値解析部2.0による解析手順について説明すると、文書実現値解析部2.0は、圧縮対象の文書実現値を最後まで走査したか否かを判断しながら（ステップS31）、文書実現値を走査し（ステップS32）、文書実現値の記述を先頭から順次認識し、「<」が記述されているか否かを調べていく（ステップS33）。なお、「<」は、XMLの仕様上、文書実現値の内容には記述されない。

【0123】文書実現値の記述として「<」が検出された場合（ステップS33のYESルート）、「<」に続く1バイトの記述に基づいて、この「<」で始まるタグが開始タグか終了タグかを判定する（ステップS4

4）。その判定は、「<」に続く記述が「/」であるか否かによって行なわれる。即ち、「<」に続く記述が「/」である場合、そのタグは終了タグであると判定され、「<」に続く記述が「/」ではない場合、そのタグは開始タグであると判定される。

【0124】開始タグの場合（ステップS34のYESルート）、その開始タグ内に記述されている要素名や属性名を検出し、それぞれ、要素名一覧表および属性名一覧表に登録する（ステップS35、S36）。その際、要素名や属性名の出現頻度も集計する。この出現頻度は、第2実施形態で必要となるタグ辞書を作成する際に利用されるものである。なお、開始タグ内には、属性名が記述されていない場合があるが、その場合、属性名は検出されないので、ステップS36の処理は省略される。一覧表への登録を終了した後は、ステップS31へ戻る。

【0125】一方、終了タグの場合（ステップS34のNOルート）、その終了タグ内に記述されている要素名を検出し（ステップS37）、その要素名が、要素名一覧表において最後に登録された要素名と一致するか否かを判定する（ステップS38）。このとき、その終了タグで括られた要素（以下、注目要素と呼ぶ）の文書実現値の内容に、子要素の記述が存在する場合、終了タグ内の要素名と要素名一覧表の最後の要素名とは一致しない。また、注目要素の文書実現値の内容に子要素の記述が存在しない場合、即ち、注目要素が葉の要素である場合、終了タグ内の要素名と要素名一覧表の最後の要素名とは一致する。

【0126】従って、ステップS38で要素名が一致しないと判定された場合（NOルート）、注目要素は子要素を有するものであって葉の要素ではなく、そのままステップS31へ戻る。これに対し、ステップS38で要素名が一致すると判定された場合（YESルート）、注目要素は子要素を有しない葉の要素であると判断することができ、統合して、その注目要素の内容中に外部ファイルに対する実体参照が記述されているか否かを判定する（ステップS39）。

【0127】葉要素の内容は、第1実施形態の圧縮変換により親要素の属性として取り扱われることになるが、前述した通り、XMLの仕様上、外部ファイルに対する実体参照を属性値において用いることができない。そこで、ステップS39で実体参照が記述されていると判定された場合（YESルート）、そのままステップS31へ戻る。つまり、外部ファイルに対する実体参照をもつ葉の要素は、「葉の要素一覧表」には登録されない。

【0128】一方、ステップS39で注目要素の内容に実体参照が記述されていないことが確認された場合（NOルート）、葉の要素一覧表に、注目要素の要素名が葉の要素名として登録・追加されるとともに、その葉の親の要素名も登録・追加される（ステップS4

0)。この後、ステップS31に戻る。

【0129】そして、ステップS31において圧縮対象の文書実現値を最後まで走査したと判定された場合(YESルート)、文書実現値の走査中に出現した要素名および属性名の出現頻度に基づいて、出現頻度の高い要素名や属性名を、より短い文字列(例えば1バイト;短縮文字列)に対応させるタグ辞書(図14の符号90参照)を作成・出力するとともに(ステップS41、S42)、最終的に得られた「葉の要素一覧表」(例えば図2(D)参照)を出力して(ステップS43)、処理を終了する。

【0130】なお、第1実施形態では、文書実現値解析部20によりタグ辞書を作成しているが、このタグ辞書は、第1実施形態の圧縮手法を実行する際には用いられず、後述する第2実施形態において用いられるものである。従って、第1実施形態では、ステップS41およびS42を省略してもよい。また、第2実施形態では、ステップS41およびS42の処理は、文書実現値解析部20ではなく、タグ辞書作成部80(図14参照)により実行されるものとして説明される。

【0131】次に、図7に示すフローチャート(ステップS51～S58)に従って、第1実施形態のDTD解析部30による解析手順について説明すると、DTD解析部30は、構成変更対象のDTDを最後まで走査したか否かを判断しながら(ステップS51)、DTDを走査し(ステップS52)、DTDの記述を先頭から順次認識し、「<ELEMENT>」が記述されているか否かを調べていく(ステップS53)。

【0132】例えば図4(A)の2行目に示すごとく、要素型宣言では、「<ELEMENT>」の後に要素名および内容モデル(子要素の要素名)が記述される。内容モデル内において、「#PCDATA」のような子約語のみが記述され、独自の子の要素名が登録されていない場合、その要素型宣言は、葉の要素を対象としたものということになる。

【0133】そこで、ステップS53でDTDの記述として「<ELEMENT>」が検出された場合(YESルート)、「!<ELEMENT>」に続く要素文字列(要素名)を検出してから(ステップS54)、さらにその後に統いて記述される内容モデルの記述を調査し(ステップS55)、内容モデル内に子の要素名が記述されているか否かを判定する(ステップS56)。

【0134】内容モデル内に要素名が記述されていない場合(ステップS56のNOルート)、今注目している要素型宣言は、葉の要素にかかるものであると判断され、その要素型宣言内の要素名(ステップS54で検出したもの)を、親の要素名とともに「葉の要素一覧表」に登録してから(ステップS57)、ステップS51に戻る。

【0135】ステップS53でDTDの記述として「!

<ELEMENT>」が検出されなかった場合(NOルート)や、ステップS56で内容モデル内に子の要素名が記述されていると判定された場合(YESルート)には、「葉の要素一覧表」への登録処理を行なうことなく、ステップS51に戻る。そして、ステップS51において構成変更対象のDTDを最後まで走査したと判定された場合(YESルート)、最終的に得られた「葉の要素一覧表」(例えば図2(D)参照)を出力して(ステップS58)、処理を終了する。

【0136】ただし、DTDにおいて内容モデルにより内容の型(例えば#PCDATA)が定義されていても、そのDTDの記述からは、内容に実体参照が含まれるか否かを認識することはできない。つまり、DTD解析部30は、前述した文書実現値解析部20とは異なり、DTDを解析しただけでは、そのDTDに従って記述される文書実現値の内容に実体参照が含まれるか否かを認識することはできず、当然、その実体参照が文書内を対象とするものか外部ファイルを対象とするものかを区別することはできない。

【0137】図8に示すフローチャート(ステップS61～S72)に従って、第1実施形態の文書実現値構成変更部40による構成変更手順について説明すると、文書実現値構成変更部40は、まず、文書実現値解析部20やDTD解析部30で得られた「葉の要素一覧表」を入力してから(ステップS61)、圧縮対象の文書実現値を最後まで走査したか否かを判断しながら(ステップS62)、文書実現値を走査する(ステップS63)。

【0138】その後、文書実現値の記述を、「葉の要素一覧表」に登録された葉の要素名と比較しながら、先頭から順次認識し、その文書実現値の記述が、「葉の要素一覧表」に登録された葉の要素であるか否かを判断する(ステップS64)。「葉の要素一覧表」に登録された葉の要素を文書実現値中で検出した場合(YESルート)、その葉の要素が属性を有しているか否かを判定し(ステップS65)、属性を有している場合(YESルート)には、その属性、つまり属性名および属性値の文字列をそれぞれ属性名一覧および属性値一覧に登録する(ステップS66)。

【0139】属性情報を有していない場合(ステップS65のNOルート)や、ステップS66での登録処理の終了後には、その葉の要素名および内容の文字列をそれぞれ属性名一覧および属性値一覧に登録する(ステップS67)。そして、ステップS66やS67による登録処理を完了した葉の要素についての、開始タグ、内容および終了タグを、文書実現値から削除してから(ステップS68)、ステップS62へ戻る。

【0140】また、ステップS64で葉の要素が検出されなかった場合(NOルート)の場合は、ステップS65～S68の処理を行なうことなくステップS62へ戻る。ステップS62において圧縮対象の文書実現値を最

後で走査したと判定された場合(YESルート)、「葉の要素一覧表」から葉の親要素を検出し(ステップS69)、その親要素の開始タグに、属性名一覧および属性値一覧にそれぞれ登録されている属性名および属性値を新たに附加する(ステップS70)。

【0141】この後、親要素の終了タグ“<親の要素名/”を削除してから(ステップS71)、親の開始タグの最後に記述された“>”の前に、“/”を記入することにより、葉の親要素の開始タグを空要素タグに変更して(ステップS72)、処理を終了する。

【0142】図9に示すフローチャート(ステップS81～S90)に従って、第1実施形態のDTD構成変更部50において構成変更手順について説明すると、DTD構成変更部50は、まず、文書実現値解説部20やDTD解説部30で得られた「葉の要素一覧表」を入力してから(ステップS81)、構成変更対象のDTDを最後まで走査したか否かを判断しながら(ステップS82)、DTDを走査する(ステップS83)。

【0143】その後、「葉の要素一覧表」に登録された葉の要素名を有する要素型宣言、即ち“<ELEMENT 葉の要素名>”が記述されているか否かを判断する(ステップS84)。そのような葉の要素型宣言が記述されている場合(ステップS84のYESルート)、その葉の要素型宣言をDTDから削除した後(ステップS85)、その葉の要素名を有する属性リスト宣言、つまり“<!ELEMENT 葉の要素名 属性名>”が記述されているか否かを判断する(ステップS86)。

【0144】そのような葉の属性リスト宣言が記述されている場合(ステップS86のYESルート)、その葉の属性リスト宣言をDTDから削除した後(ステップS87)、その葉についての親要素の要素型宣言における内容モデルの記述から葉(子要素)の記述を削除する(ステップS88)。

【0145】そして、葉の親要素についての属性リスト宣言において、ステップS85で削除した葉の要素についての要素名および内容を、それぞれ新たな属性名および属性値として付加するとともに、ステップS87で削除した葉の要素についての属性名および属性値を、それそれぞれ新たな属性名および属性値として付加してから(ステップS89)、ステップS82へ戻る。このとき、親要素についての属性リスト宣言が、構成変更前に存在していない場合には、新たに属性リスト宣言を作成する。

【0146】なお、ステップS84で葉の要素型宣言が記述されていないと判断された場合(NOルート)、ステップS82へ戻る。また、ステップS86で葉の属性リスト宣言が記述されていないと判断された場合(NOルート)、ステップS88へ移行する。ステップS86のNOルートもしくはステップS87からステップS88へ移行した時に、既に内容モデルから葉の記述が削除されている場合には、ステップS88では何ら処理を行

なうことなく、ステップS89へ移行する。

【0147】さらに、ステップS86のNOルートを経由してステップS88へ移行した場合には、葉の親要素についての属性リスト宣言において、ステップS85で削除した葉の要素についての要素名および内容を、それぞれ新たな属性名および属性値として付加してから、ステップS82へ戻る。このときも、親要素についての属性リスト宣言が、構成変更前に存在していない場合には、新たに属性リスト宣言を作成する。

【0148】このように、本発明の第1実施形態によれば、文書実現値における要素の木構造を解析し、その解析結果に従って、葉要素についての開始タグ、終了タグおよび内容を文書実現値から削除し、その葉要素の要素名、属性名および属性値を親要素の属性として親要素の開始タグ内に付加することにより、葉要素にかかる記述を親要素の属性として取り扱うことができ、葉要素の開始タグや終了タグを記述する必要がなくなり、XML文書の特徴を損なうことなく、また、検索可能な状態に保持したまま、葉要素にかかるタグの記述が省略・圧縮される。

【0149】従って、XML文書の圧縮率を大幅に高めることができ、ひいては、大規模なデータベースを取り扱うシステムにおいて文書データの格納効率を大幅に高めることができる。特に、多数の短い語句をもつ品目表や価格表等をXML文書で記述するような場合、短い語句(内容)を挟んだ開始タグと終了タグとの対表現を省略することができるので、その圧縮率を大幅に高めることができる。

【0150】このとき、圧縮後も、データ長の短い内容の平文を検索対象として扱うことが可能であり、検索を行なう際にはXML文書を復元(伸長)する必要がない。また、XML文書の特徴を損なわないため、ブラウザなどの応用ソフトウェアとの整合を容易にとることができる。さらに、親要素の終了タグを削除して親要素の開始タグを空要素タグに変更することで、XML文書の圧縮率をより高めることができる。

【0151】同様に、DTDにおける要素の木構造を解析し、その解析結果に従って、葉要素の要素型宣言や属性リスト宣言をDTDから削除するとともに葉要素にかかる記述を親要素の要素型宣言(内容モデル)から削除し、その葉要素の要素型宣言や属性リスト宣言にかかる情報を親要素の属性として再定義することにより、文書実現値に対して行なわれた圧縮に対応した圧縮処理がDTDに対しても行なわれ、葉要素にかかる記述を親要素の属性として取り扱うことができる。従って、XML文書の特徴を損なうことなく、また、検索可能な状態に保持したまま、葉要素にかかる要素型宣言や属性リスト宣言の記述が省略されてDTDが圧縮されるので、XML文書の圧縮率をより高めることができる。

【0152】〔2〕第2実施形態の説明

次に、本発明の第2実施形態について説明する。まず、図15 (A)～図15 (D) を参照しながら、本発明の第2実施形態における構造化文書の圧縮原理を説明する。なお、本発明の第2実施形態でも、構造化文書がXML文書である場合について説明する。

【0153】第2実施形態では、文書実現値のタグ内やDTDにおける要素名および属性名の各文字列を、1または2バイトの文字列に置換し、その対応関係をタグ辞書(図14の符号90参照)に記録する。通常、タグ内に記述される文字列(要素名や属性名)は、人が読んで意味が分かるように数バイト以上の長さの文字列を用いて、DTDで定義されている。

【0154】ただし、要素名および属性名の先頭文字は、SGMLでは1バイトの英字(A～z, a～z)に限られる。一方、XMLでは、先頭文字は、1バイトの英字、2バイトの平仮名またはカタカナ、1バイトの

“#”または“:”のいずれかに限られる。一般に、文書実現値のタグ部分だけで、すべての文書量の6割から8割が占められる。このため、タグ内における文字列の可読性を犠牲にして、その文字列を1または2バイトの文字列に置換するだけで、XML文書の圧縮率を大幅に高めることができる。

【0155】そこで、本発明の第2実施形態では、図15 (A)～図15 (D) に示すように、要素名および属性名の既存の名前と、新たに定義した1または2バイトの短縮文字列との間の対応関係をタグ辞書に記録し、そのタグ辞書に基づいて、文書実現値のタグ内およびDTDにおける該当する文字列を、より短い短縮文字列に置き換える。この短縮文字列は、当然、既存の名前の文字列よりも短く且つその文字列を特定するものでなければならない。

【0156】図15 (A) はタグの具体的な記述例を示す図で、この図15 (A) に示すタグでは、要素名「title」の要素に対して、属性名「tspirit」および属性値「スクールC A T シリーズNO.563 3(3)」をもつ属性が付与されている。このとき、図15 (B) に示すとく、要素名「title」の置換文字(短縮文字列)として「a」を予め設定してタグ辞書に登録しておくとともに、図15 (C) に示すとく、属性名「tspirit」の置換文字(短縮文字列)として「A」を予め設定してタグ辞書に登録しておく。

【0157】そして、図15 (B) や図15 (C) に示すタグ辞書を用いることにより、図15 (A) に示すタグにおいて、要素名および属性名の文字列を、図15 (D) に示すように、1バイトの文字列に置き換える。このとき、DTDを有する検証済みXML文書(パターン⑩、⑪)においては、そのDTDも、上述した文字列置換に対応して変換される。

【0158】従って、SGMLバーサーやXMLバーサー(プロセッサ)等では、上述のごとく変換されたDTD

に基づいて、同じく上述のごとく変換された構造化文書が解析される。ただし、応用ソフトウェア側で要素名や属性の探索を行なう際には、変換されたDTDから読み取った1または2バイトの短縮文字列を用いて、要素名および属性名を指定しなければならない。

【0159】以下、図14および図16～図24を参照しながら、本発明の第2実施形態について、より詳細かつ具体的に説明する。図14は本発明の第2実施形態としての構造化文書の圧縮装置の機能構成を示すブロック図であり、この図14に示すように、第2実施形態の圧縮装置は、第1実施形態と同様の文書記憶部10、文書実現値解析部20、DTD解析部30、新規DTDファイル作成部60および新旧DTD対応表出力部70のほかに、タグ辞書作成部80、タグ辞書90、文書実現値文字列置換部41およびDTD文字列置換部51を有して構成されている。

【0160】ここで、第2実施形態の圧縮装置も、第1実施形態と同様、CPU、RAM、ROMなどをバスラインにより接続して構成される、パソコン等のコンピュータシステムにより実現されるものである。つまり、RAMやROMが文書記憶部10としての機能を果たすほか、RAMには、文書実現値解析部20、DTD解析部30、新規DTDファイル作成部60、新旧DTD対応表出力部70、タグ辞書作成部80、文書実現値文字列置換部41およびDTD文字列置換部51を実現するためのアプリケーションプログラムが格納されている。また、タグ辞書90は、例えばRAM上に記録・保存され

る。【0161】そして、CPUが、上記アプリケーションプログラムを実行することにより、文書実現値解析部20、DTD解析部30、新規DTDファイル作成部60、新旧DTD対応表出力部70、タグ辞書作成部80、文書実現値文字列置換部41およびDTD文字列置換部51を実現するためのアプリケーションプログラムが格納されている。また、タグ辞書90は、例えばRAM上に記録・保存され

るようになっている。

【0162】この第2実施形態の圧縮装置を実現するためのプログラムも、第1実施形態と同様、例えばフレキシブルディスク、CD-ROM等の、コンピュータ読取可能な記録媒体に記録された形態で提供される。そして、コンピュータはその記録媒体からプログラムを読み取って内部記憶装置または外部記憶装置に転送し格納して用いる。また、そのプログラムを、例えば磁気ディスク、光ディスク、光磁気ディスク等の記憶装置(記録媒体)に記録しておき、その記憶装置から通信経路を介してコンピュータに提供してもよい。

【0163】そして、第2実施形態の圧縮装置としての機能をコンピュータにより実現する際には、内部記憶装置(例えばRAM)に格納された上記プログラムがコンピュータのマイクロプロセッサ(例えばCPU)によっ

て実行される。このとき、記録媒体に記録されたプログラムをマイクロプロセッサが直接読み取って実行してもよい。

【0164】さて、図14に示す第2実施形態の圧縮装置において、文書記憶部10、文書実現値解析部20、DTD解析部30、新規DTDファイル作成部60および新旧DTD対応表出力部70は、第1実施形態で説明したものとほぼ同様の機能を果たすので、その詳細な説明は省略する。ただし、第1実施形態の文書実現値解析部20は、第2実施形態のタグ辞書作成部80としての機能を有していたが、第2実施形態では、図6のステップS41、S42に対応した処理を行なう部分を、タグ辞書作成部80として、文書実現値解析部20から機能的に分離して説明する。

【0165】このタグ辞書作成部80は、文書実現値解析部20やDTD解析部30による解析結果に従い、文書実現値のタグ内およびDTDに記述された文字列（要素名、属性名）とその文字列よりも短く且つその文字列を特定しうる短縮文字列（前述した1または2バイトの文字列）とを対応させるタグ辞書90を作成するものである。

【0166】なお、本実施形態のタグ辞書作成部80は、図6を参照しながら第1実施形態で説明したことなく、文書実現値解析部20による解析結果のみを用いてタグ辞書90を作成するものとなっているが、文書実現値解析部20による解析結果に従いてDTD解析部30による解析結果を用いてタグ辞書90を作成してもよいし、文書実現値解析部20による解析結果とDTD解析部30による解析結果との両方を用いてタグ辞書90を作成してもよい。

【0167】文書実現値文字列置換部41は、タグ辞書90を用いて、文書実現値のタグ内に記述された文字列（要素名、属性名）を、その文字列に対応する短縮文字列に置き換えるもので、その置換手順については、図7に示すフローチャートを参照しながら後述する。

【0168】DTD文字列置換部51は、XML文書が検証済みXML文書である場合（つまりパターン④または⑤のXML文書である場合）に、DTDの記述を、文書実現値文字列置換部41によって置換された文書実現値の記述に合わせるべく、タグ辞書90を用いて、DTDに記述された文字列（要素名、属性名）を、その文字列に対応する短縮文字列に置き換えるもので、その置換手順については、図8に示すフローチャートを参照しながら後述する。

【0169】なお、DTD文字列置換部51において、XML文書がパターン④である場合、DTDは文書記憶部から読み込まれるが、XML文書がパターン⑤である場合、DTDは外部ファイル100から読み込まれる。また、第2実施形態でも、XML文書がパターン④である場合、DTD文字列置換部51による置換結果（圧縮

後のDTD）を、文書実現値置換部41による置換結果（圧縮後の文書実現値）とともに文書記憶部10に出力・格納しているが、圧縮後の文書実現値とともに他の記録媒体等に出力・格納してもよい。

【0170】新規DTDファイル作成部60は、DTDが外部ファイル100に存在する場合（XML文書がパターン④である場合）、DTD文字列置換部51により置換処理されたDTDについてのファイル（新規DTDファイル）を作成して外部ファイル100へ出力するものである。新旧DTD対応表出力部70は、DTDが外部ファイル100に存在する場合（XML文書がパターン④である場合）、置換処理前のDTDと置換処理後の新規DTDとの対応関係を明記した新旧DTD対応表（例えば図24（G）参照）を作成して文書記憶部10へ出力するものである。

【0171】上述のごとく第2実施形態の圧縮手法（文字列の置換）により圧縮されたXML文書は、XML文書としての特徴を全く損なっておらず、圧縮状態のままで（伸長することなく）XML文書としての機能を果たすことができる。このとき、タグ辞書90を保持しておけば、このタグ辞書90を参照して置換前の文字列と短縮文字列との対応関係を認識することにより、文書実現値やDTDにおいて置換・圧縮された文字列を伸長することなく、XML文書内のデータを検索することができる。

【0172】なお、上述のごとく短縮文字列に置換されたXML文書の記述を元の状態に伸長・復元させるために、文書実現値文字列逆置換手段やDTD文字列逆置換手段を含んで構成された伸長装置（図示省略）をそなえておく。ここで、文書実現値文字列逆置換手段は、上述した文書実現値文字列置換部41とは逆の置換処理を行なうもので、タグ辞書90を用いて、文書実現値のタグ内に記述された短縮文字列を、元の文字列（要素名、属性名）に置き換えるものであり、DTD文字列逆置換手段は、上述したDTD文字列置換部51とは逆の置換処理を行なうもので、タグ辞書90を用いて、DTDに記述された短縮文字列と、元の文字列（要素名、属性名）に置き換えるものである。

【0173】次に、図16～図24を参照しながら、第2実施形態について説明する。まず、図16に示すフローチャート（ステップS111～S129）に従い、第2実施形態における構造化文書（XML文書）の圧縮手順を説明する。なお、図14では図示省略しているが、第2実施形態の圧縮装置にも、文書記憶部10に保存されているXML文書がパターン①～⑤（表2参照）のいずれのものであるからを認識するためのパターン認識機能がそなえられている。このパターン認識機能による処理は、図16に示すステップS112～S114による処理に対応している。

【0174】圧縮対象のXML文書が入力され文書記憶

部10に格納されると（ステップS111）、そのXML文書に“<!DOCTYPE”が記述されているか否かを判定し（ステップS112）、記述されていない場合（ステップS112のNOルート）、そのXML文書はDTDをもたない整形式XML文書、つまりパターン⑩のXML文書であると認識され、後述するごとくステップS115、S116およびS129が実行される。

【0175】XML文書に“<!DOCTYPE”が記述されている場合（ステップS112のYESルート）、その後に“[”が記述されているか否かを判定する（ステップS113）。“<!DOCTYPE”は記述されているが“[”が記述されていない場合（ステップS113のNOルート）、そのXML文書は、DTDを外部ファイル100として有する検証済みXML文書、つまりパターン⑤のXML文書であると認識され、後述するごとくステップS121～S129が実行される。

【0176】“[”が記述されている場合（ステップS113のYESルート）、“<!ELEMENT”（もしくは“<!ATTLIST”）が記述されているか否かを判定する（ステップS114）。“<!DOCTYPE”および“[”は記述されているが“<!ELEMENT”が記述されていない場合（ステップS114のNOルート）、内部または外部への実体宣言を含むDTDを有する整形式XML文書、つまりパターン⑨または⑩のXML文書であると認識され、パターン⑩の場合と同様、ステップS115、S116およびS129が実行される。

【0177】“<!DOCTYPE”、“[”および“<!ELEMENT”がいずれも記述されている場合（ステップS114のYESルート）、XML文書内にDTDを有する検証済みXML文書、つまりパターン⑩のXML文書であると認識され、ステップS117～S120およびS129が実行される。以下、各パターン①～⑩に対する圧縮処理について、図20～図24に示す具体例（第1例～第5例）を参照しながら説明する。

【0178】図20（A）～図20（C）はいずれも第2実施形態によるXML文書の具体的な圧縮処理（第2例）を説明するための図である。図21（A）に示す圧縮前のXML文書は、パターン⑩のXML文書であり、図10（A）に示したものと同じである。この図20（A）に示すXML文書（パターン①）には“<!DOCTYPE”が記述されていないので、処理はステップS112のNOルートからステップS115へ移行し、文書実現値解釈部20によって文書実現値のタグ内の記述が解釈され、タグ辞書作成部80により、図20（C）に示すようなタグ辞書90が登録・作成される。

【0179】ここで、図20（A）に示す例では、文書実現値の各タグ内には要素名のみが記述され、どの要素も属性を有していないので、要素名だけが検出され、各

要素名に短縮文字列を対応させるタグ辞書90が登録・作成される。図20（C）に示すタグ辞書90は、文書実現値解釈部20によって検出・認識された要素名“book”、“title”、“author”に、1バイトの短い短縮文字列（以下、置換文字という場合がある）、例えば“a”、“b”、“c”をそれぞれ対応させるものとなっている。

【0180】なお、タグ内に属性名も記述されている場合には、図23（D）や図24（E）を参照しながら後述するごとく、属性名についてのタグ辞書90も登録・作成される。そして、ステップS115で得られたタグ辞書90を用いて、文書実現値文字列置換部41により、文書実現値のタグ内に記述された要素名“book”、“title”、“author”が、それぞれ、1バイトの置換文字“a”、“b”、“c”に置き換えられる（ステップS116）。

【0181】ステップS115およびS116によって、例えば図20（A）に示すXML文書は、図20（B）に示すようなXML文書に置換・圧縮されてから、圧縮文書として文書記憶部10等へ出力・格納される（ステップS129）。図20（B）に示すXML文書において、1行目のXML宣言の記述は圧縮前と変わらないが、2～5行目においては、開始タグ内および終了タグ内の要素名“book”、“title”、“author”がそれぞれ1バイトの置換文字“a”、“b”、“c”に置換されている。

【0182】図21（A）～図21（C）はいずれも第2実施形態によるXML文書の具体的な圧縮処理（第2例）を説明するための図である。図21（A）に示す圧縮前のXML文書は、パターン⑩のXML文書であり、図11（A）に示したものと同じである。この図21（A）に示すXML文書（パターン⑩）には、“<!DOCTYPE”および“[”がいずれも記述されているが、“<!ELEMENT”や“<!ATTLIST”が記述されていないので、処理はステップS114のNOルートからステップS115へ移行し、前述したパターン⑩のXML文書と同様の処理が実行される。このとき、タグ辞書作成部80により、図21（C）に示すごとく、図20（C）に示したものと同じタグ辞書90が登録・作成される。

【0183】これにより、例えば図21（A）に示すXML文書は、図21（B）に示すようなXML文書に置換・圧縮されてから、圧縮文書として文書記憶部10等へ出力・格納される（ステップS129）。図21（B）に示すXML文書において、1、3および4行目の記述は圧縮前と変わらないが、2、5～8行目におけるタグ内の要素名“book”、“title”、“author”がそれぞれ1バイトの置換文字“a”、“b”、“c”に置換される。

【0184】図22（A）～図22（C）はいずれも第

2実施形態によるXML文書の具体的な圧縮処理(第3例)を説明するための図である。図22(A)に示す圧縮前のXML文書は、前述したパターン③のXML文書であり、1行目に、この文書がバージョン1.0のXML文書であることを示すXML宣言が記述され、2~4行目に、外部への実体宣言を含むDTDが記述され、5~8行目に文書実現値が記述されている。

【0185】2~4行目のDTDでは、文書型宣言に含まれる実体宣言(3行目)におけるシステム識別子“SYSTEM”を用いて、文書実現値(XMLインスタンス)内で用いられる文字列“para”的実体として、URL “http://www.xml.co.jp”で指定される外部ファイルを用いることが宣言・定義されている。また、5~8行目に記述された文書実現値は、図10(A)に示したXML文書の2~5行目の記述例とほぼ同一であるが、図22(A)に示す例では、7行目の要素“author”的内容として、「佐藤元¶」が記述されている。

【0186】ここで、「¶」は、文字列“para”的実体を参照することを指示する記述であり、実際に表示・印刷等によって出力される文書中では、URL “http://www.xml.co.jp”で指定される外部ファイルが読み出されて表記されることになる。

【0187】そして、図22(A)に示すXML文書(パターン③)には、パターン②のXML文書と同様、“!DOCTYPE”および“[”がいずれも記述されているが、“!ELEMENT”や“!ATTLIST”が記述されていないので、処理はステップS114のNOルートからステップS115へ移行し、前述したパターン①や②のXML文書と同様の処理が実行される。このとき、タグ辞書作成部80により、図22(C)に示すごとく、図20(C)に示したものと同じタグ辞書90が登録・作成される。

【0188】これにより、例えば図22(A)に示すXML文書は、図22(B)に示すようなXML文書に置換・圧縮されてから、圧縮文書として文書記憶部10等へ出力・格納される(ステップS129)。図22(B)に示すXML文書において、1、3および4行目の記述は圧縮前と変わらないが、2、5~8行目におけるタグ内の要素名“book”, “title”, “author”がそれぞれ1バイトの置換文字“a”, “b”, “c”に置換される。

【0189】図23(A)~図23(D)はいずれも第2実施形態によるXML文書の具体的な圧縮処理(第4例)を説明するための図である。図23(A)に示す圧縮前のXML文書は、パターン④のXML文書であり、図12(A)に示したものとはほぼ同じである。ただし、図23(A)に示すXML文書では、要素“author”的属性“year”についての記述が省略されている。つまり、DTDにおいて、属性“year”についての属性リスト

ト宣言が省略されるとともに、要素“author”的開始タグ内における属性記述が省略されている。

【0190】図23(A)に示すXML文書(パターン④)には、“!DOCTYPE”および“[”が記述されるとともに、“!ELEMENT”または“!ATTLIST”も記述されているので、処理はステップS114のYESルートからステップS117へ移行し、文書実現値解析部20によって文書実現値のタグ内の記述が解釈されるとともに、DTD解析部30によってDTDの記述が解釈される(ステップS118)。

【0191】このとき、タグ辞書作成部80により、図23(C)に示すような、要素名のためのタグ辞書90と、図23(D)に示すような、属性名のためのタグ辞書90とが登録・作成される。ここで、図23(C)に示すタグ辞書90は、図20(C)に示すものと同じで、文書実現値解析部20やDTD解析部30によって検出・認識された要素名“book”, “title”, “author”に、1バイトの短縮文字列、例えば“a”,

“b”, “c”をそれぞれ対応させるものとなっている。また、図23(D)に示すタグ辞書90は、文書実現値解析部20やDTD解析部30によって検出・認識された属性名“field”に、1バイトの短縮文字列、例えば“A”を対応させるものとなっている。

【0192】そして、図23(C)および図23(D)に示すタグ辞書90を用い、文書実現値文字列置換部41により、文書実現値のタグ内に記述された要素名“book”, “title”, “author”や属性名“field”がそれぞれ1バイトの短縮文字列“a”, “b”, “c”, “A”に置き換えられるとともに(ステップS119)、文書実現値文字列置換部41による文書実現値の文字列置換に合わせ、DTD文字列置換部51により、DTDに記述された要素名“book”, “title”, “author”や属性名“field”がそれぞれ1バイトの短縮文字列“a”, “b”, “c”, “A”に置き換えられる(ステップS120)。

【0193】これにより、例えば図23(A)に示すXML文書は、図23(B)に示すようなXML文書に置換・圧縮されてから、圧縮文書として文書記憶部10等へ出力・格納される(ステップS129)。図23(B)に示すXML文書において、1および7行目の記述は圧縮前と変わらないが、2~6および8~11行目における要素名“book”, “title”, “author”や属性名“field”がそれぞれ1バイトの置換文字“a”, “b”, “c”, “A”に置換される。

【0194】図24(A)~図24(G)はいずれも第2実施形態によるXML文書の具体的な圧縮処理(第5例)を説明するための図である。図24(A)に示す圧縮前のXML文書は、パターン⑤のXML文書であり、図13(A)に示したものとはほぼ同じである。ただし、図24(A)に示すXML文書では、要素“author”的

属性「year」についての記述が省略されている。つまり、DTDにおいて、属性「year」についての底性リスト宣言が省略されるとともに、要素「author」の開始タグ内における属性記述が省略されている。

【0195】2行目のDTDの文書型宣言では、システム識別子“SYSTEM”により、外部ファイル100に保持されたDTD（ファイル名「..vbook.dtd」）を用いることが宣言・定義されている。そして、ファイル名「..vbook.dtd」のDTDは、図24（A）における文書実現値の構成に対応して、図24（B）に示すよう記述されている。この図24（B）に示すDTD（1～4行目）は、図4（A）に示したDTDにおける2～5行目の記述例と同一であるので、その説明は省略する。

【0196】図24（A）に示すXML文書（パターン⑩）には、“<!DOCTYPE”は記述されているが、その後には“（”が記述されることなく、外部ファイル100におけるDTDを指定するシステム識別子が記述されているので、処理はステップS113のNOLートからステップS121へ移行し、文書実現値解析部20によって文書実現値のタグ内の記述が解析されるとともに、DTD解析部30によって、システム識別子に従って外部ファイル100から読み込まれたDTD（ファイル名「..vbook.dtd」）の記述が解析される（ステップS122）。

【0197】このとき、タグ辞書作成部80により、図24（E）に示すような、要素名のためのタグ辞書90と、図24（F）に示すような、属性名のためのタグ辞書90とが登録・作成される。ここで、図24（E）に示すタグ辞書90は、図20（C）に示すものと同じであり、図24（F）に示すタグ辞書90は、図23（D）に示すものと同じである。

【0198】この後、図24（B）に示すDTDを変更・圧縮して得られる新規のDTDのために、元のファイル名とは異なる新規のファイル名（例えば「..vbook2.dtd」）を設定して文書実現値に記入することにより、文書実現値における文書型宣言のシステム識別子“SYSTEM”により指定されるファイル名を、旧ファイル名「..vbook.dtd」から、新規ファイル名「..vbook2.dtd」に書き換える。

【0199】そして、図24（E）および図24（F）に示すタグ辞書90を用い、文書実現値文字列置換部41により、文書実現値のタグ内に記述された要素名「book」、「title」、「author」や属性名「field」がそれぞれ1バイトの短縮文字列「a」、「b」、「c」、「」、「A」に置き換える（ステップS124）。

【0200】これにより、図24（A）に示すXML文書は、図24（C）に示すようなXML文書に置換・圧縮される。図13（C）に示すXML文書において、1行目の記述は圧縮前と変わらないが、2行目のシステム

識別子“SYSTEM”により指定されるファイル名が新規ファイル名「..vbook2.dtd」となるとともに、2～6行目における要素名「book」、「title」、「author」や属性名「field」がそれぞれ1バイトの置換文字「a」、「b」、「c」、「A」に置き換される。

【0201】ついで、新規DTDファイルを作成部60により、新規のDTDファイルを作成し、そのDTDファイルに、外部ファイル100から読み込んだ圧縮前のDTDファイルの内容を複写してから（ステップS125）、文書実現値文字列置換部41による文書実現値の文字列置換に合わせ、DTD文字列置換部51により、DTDに記述された要素名「book」、「title」、「author」や属性名「field」がそれぞれ1バイトの短縮文字列「a」、「b」、「c」、「A」に置き換えられる（ステップS126）。

【0202】これにより、図24（B）に示すDTDは、図24（D）に示すようなDTDに変更・圧縮される。図24（D）に示すDTDでは、1～4行目における要素名「book」、「title」、「author」や属性名「field」がそれぞれ1バイトの文字列「a」、「b」、「c」、「」、「A」に置き換される。そして、DTD文字列置換部51で置換・圧縮されたDTDのファイル（新規DTDファイル）は、新規のファイル名「..vbook2.dtd」を付与されて、新規DTDファイル作成部60から外部ファイル100へ出力・格納される（ステップS127）。

【0203】また、新旧DTD対応表出力部70によって、IBD TDと新規DTDとの対応関係（具体的には旧ファイル名と新規ファイル名との対応関係）を記述した新旧DTD対応表が、図24（G）に示すように作成され、文書記憶部10等へ出力・格納されるとともに（ステップS128）、ステップS124において置換・圧縮されたXML文書は、圧縮文書として文書記憶部10等へ出力・格納される（ステップS129）。その際、タグ辞書90や新旧DTD対応表は、独立したファイルではなく、圧縮文書に注釈の形で付加してもよい。

【0204】さて、次に、図17および図18を参照しながら、第2実施形態の圧縮装置を構成する文書実現値文字列置換部41およびDTD文字列置換部51の動作について説明する。まず、図17に示すフローチャート（ステップS151～S158）に従って、第2実施形態の文書実現値文字列置換部41による置換手順について説明すると、文書実現値文字列置換部41は、まず、タグ辞書作成部80で得られたタグ辞書90を入力してから（ステップS151）、第1実施形態の文書実現値解析部20と同様にして（図6のステップS31～S34参照）、文書実現値における開始タグおよび終了タグを判別し（ステップS152～S155）、それらのタグ中の要素名、属性名を、タグ辞書90を用いて置換する（ステップS156～S158）。

【0205】つまり、圧縮対象の文書実現値を最後まで走査したか否かを判断しながら（ステップS152）、文書実現値を走査し（ステップS153）、文書実現値の記述を先頭から順次認識し、“<”が記述されているか否かを調べていく（ステップS154）。なお、“<”は、XMLの仕様上、文書実現値の内容には記述されない。

【0206】文書実現値の記述として“<”が検出された場合（ステップS154のYESルート）、“<”に続く1バイトの記述に基づいて、この“<”で始まるタグが開始タグか終了タグかを判定する（ステップS155）。その判定は、“<”に続く記述が“/”であるか否かによって行なわれる。即ち、“<”に続く記述が“/”である場合、そのタグは終了タグであると判定され、“<”に続く記述が“/”ではない場合、そのタグは開始タグであると判定される。

【0207】開始タグの場合（ステップS155のYE Sルート）、タグ辞書90を参照して、その開始タグ内に記述されている要素名を、対応する短縮文字列に置き換える（ステップS156）。また、その開始タグ内に属性名が記述されている場合には、その属性名についても、タグ辞書90を参照して、対応する短縮文字列に置き換える（ステップS157）。なお、開始タグ内には、属性名が記述されていない場合には、ステップS157の処理は省略される。このような置換処理を終了した後は、ステップS152へ戻る。

【0208】一方、終了タグの場合（ステップS155のNOルート）、タグ辞書90を参照して、その終了タグ内に記述されている要素名を、対応する短縮文字列に置き換えてから（ステップS158）、ステップS152へ戻る。そして、ステップS152において、圧縮対象の文書実現値を最後まで走査したと判定された場合（YESルート）、処理を終了する。

【0209】図18に示すフローチャート（ステップS161～S170）に従って、第2実施形態のDTD文字列置換部51による置換手順について説明すると、DTD文字列置換部51は、まず、タグ辞書作成部80で得られたタグ辞書90を入力してから（ステップS161）、圧縮対象のDTDを最後まで走査したか否かを判断しながら（ステップS162）、DTDを走査し（ステップS163）、要素型宣言、即ち“<!ELEMENT”が記述されているか否かを調べていく（ステップS164）。

【0210】ステップS164で“<!ELEMENT”が検出された場合（YE Sルート）、タグ辞書90を参照して、その要素型宣言内の要素名を、対応する短縮文字列に置き換えてから（ステップS165）、その要素型宣言内に記述された内容モデルを検出し（ステップS166）、その内容モデルに子の要素名が記述されている場合には、タグ辞書90を参照してその要素名に

ついで、対応する短縮文字列に置き換える（ステップS167）。なお、内容モデルの記述が無い場合や、内容モデルに子の要素名が記述されていない場合には、ステップS167の処理は省略される。

【0211】この後、属性リスト宣言、つまり“<!ATT LIST”が記述されているか否かを調べる（ステップS168）。ステップS168で“<!ATT LIST”が検出された場合（YESルート）、タグ辞書90を参照して、その属性リスト宣言内の要素名を、対応する短縮文字列に置き換えるとともに（ステップS169）、その属性リスト宣言内の属性名を、対応する短縮文字列に置き換えてから（ステップS170）、ステップS162へ戻る。

【0212】なお、ステップS164で要素型宣言が記述されていないと判断された場合（NOルート）や、ステップS168で属性リスト宣言が記述されていないと判断された場合（NOルート）には、ステップS162へ戻る。そして、ステップS152において、圧縮対象の文書実現値を最後まで走査したと判定された場合（YESルート）、処理を終了する。

【0213】ところで、前述したように、第2実施形態では、短縮文字列に置換されたXML文書の記述を元の状態に伸長・復元させるための伸長装置（図示省略）がそなえられている。この伸長装置を構成する前記文字実現的文字列置換手段および前記DTD文字列置換手段は、それぞれ、文字実現文字列置換部41やDTD文字列置換部51と同様、図17および図18に示すフローチャートに従って逆置換処理を行なうものである。ただし、その逆置換処理では、図17のステップS156～S158および図18のステップS165、S167、S169およびS170における文字列変換方向が逆方向になる。

【0214】次に、図19に示すフローチャート（ステップS131～S142）に従って、第2実施形態における構造化文書の伸長手順、つまり上述した伸長装置による逆置換手順について説明する。なお、第2実施形態の伸長装置にも、伸長すべき圧縮文書がパターン①～⑩のいずれのものであるからを認識するためのパターン認識機能がそなえられている。このパターン認識機能による処理は、図19に示すステップS133～S135による処理に対応している。

【0215】まず、伸長対象の圧縮文書が入力されるとともに（ステップS131）、圧縮処理時に作成されたタグ辞書90が入力されると（ステップS132）、その圧縮文書は“<!DOCTYPE”が記述されているか否かを判定し（ステップS133）、記述されていない場合（ステップS133のNOルート）、その圧縮文書はDTDをもたない整形式XML文書、つまりパターン⑩のXML文書であると認識され、後述するごくステップS136およびS142が実行される。

【0216】圧縮文書に“**<!DOCTYPE**”が記述されている場合（ステップS133のYESルート）、その後に“[”が記述されているか否かを判定する（ステップS134）。“<!DOCTYPE”は記述されているが“[”が記述されていない場合（ステップS134のNOルート）、その圧縮文書は、DTDを外部ファイル100として有する検証済みXML文書、つまりパターン⑩のXML文書であると認識され、後述するごくステップS139～S142が実行される。

【0217】“[”が記述されている場合（ステップS134のYESルート），“<!ELEMENT”（もしくは“<!ATTLIST”）が記述されているか否かを判定する（ステップS135）。“<!DOCTYPE”および“[”は記述されているが“<!ELEMENT”が記述されていない場合（ステップS135のNOルート）、その圧縮文書は、内部または外部への実体宣言を含むDTDを有する検証済みXML文書、つまりパターン⑩のXML文書であると認識され、パターン⑩の場合と同様、ステップS136およびS142が実行される。

【0218】“<!DOCTYPE”，“[”および“<!ELEMENT”がいずれも記述されている場合（ステップS135のYESルート）、その圧縮文書は、XML文書内にDTDを有する検証済みXML文書、つまりパターン⑩のXML文書であると認識され、ステップS137、S138およびS142が実行される。以下、各パターン①～⑩に対する伸長処理について、図20～図24に示す具体例（第1例～第5例）を参照しながら説明する。

【0219】伸長対象の圧縮文書が図20（B）に示すようなパターン①の圧縮XML文書である場合、その文書には“<!DOCTYPE”が記述されていないので、処理はステップS133のNOルートからステップS136へ移行し、図20（C）に示すタグ辞書90を用いて、前記文書実現値文字列逆置換手段により、文書実現値のタグ内に記述された1バイトの短縮文字列「a」、「b」、「c」が、それぞれ元の要素名「book」、「title」、「author」に逆置換される。これにより、例えば図20（B）に示す圧縮XML文書は、図20（A）に示すようなXML文書に逆置換・伸長され、伸長文書として文書記憶部10等へ出力・格納される（ステップS142）。

【0220】伸長対象の圧縮文書が図21（B）に示すようなパターン②の圧縮XML文書である場合、その文書には、“<!DOCTYPE”および“[”がいずれも記述されているが、“<!ELEMENT”や“<!ATTLIST”が記述されていないので、処理はステップS135のNOルートからステップS136へ移行し、図21（C）に示すタグ辞書90を用いて、前記文書実現値文字列逆置換手段により、文書実現値のタグ内に記述された1バイトの短縮文字列「a」、「b」、「c」や「A」が、それぞれ元の要素名「book」、「title」、「author」や属性名「field」に逆置換される。

述された1バイトの短縮文字列「a」、「b」、「c」が、それぞれ元の要素名「book」、「title」、「author」に逆置換される。これにより、例えば図21（B）に示す圧縮XML文書は、図21（A）に示すようなXML文書に逆置換・伸長され、伸長文書として文書記憶部10等へ出力・格納される（ステップS142）。

【0221】伸長対象の圧縮文書が図22（B）に示すようなパターン③の圧縮XML文書である場合、その文書には、パターン②と同様、“<!DOCTYPE”および“[”がいずれも記述されているが、“<!ELEMENT”や“<!ATTLIST”が記述されていないので、処理はステップS135のNOルートからステップS136へ移行し、前述したパターン①や②の圧縮XML文書と同様の処理が実行される。

【0222】つまり、図22（C）に示すタグ辞書90を用いて、前記文書実現値文字列逆置換手段により、文書実現値のタグ内に記述された1バイトの短縮文字列「a」、「b」、「c」が、それぞれ元の要素名「book」、「title」、「author」に逆置換される。これにより、例えば図22（B）に示す圧縮XML文書は、図22（A）に示すようなXML文書に逆置換・伸長され、伸長文書として文書記憶部10等へ出力・格納される（ステップS142）。

【0223】伸長対象の圧縮文書が図23（B）に示すようなパターン④の圧縮XML文書である場合、その文書には、“<!DOCTYPE”および“[”が記述されるとともに“<!ELEMENT”または“<!ATTLIST”も記述されているので、処理はステップS135のYESルートからステップS137へ移行し、図23（C）および図23（D）に示すタグ辞書90を用いて、前記文書実現値文字列逆置換手段により、文書実現値のタグ内に記述された1バイトの短縮文字列「a」、「b」、「c」や「A」が、それぞれ元の要素名「book」、「title」、「author」や属性名「field」に逆置換される。

【0224】さらに、図23（C）および図23（D）に示すタグ辞書90を用いて、前記DTD文字列逆置換手段により、DTDに記述された1バイトの短縮文字列「a」、「b」、「c」や「A」が、それぞれ元の要素名「book」、「title」、「author」や属性名「field」に逆置換される（ステップS138）。これにより、例えば図23（B）に示す圧縮XML文書は、図23（A）に示すようなXML文書に逆置換・伸長され、伸長文書として文書記憶部10等へ出力・格納される（ステップS142）。

【0225】伸長対象の圧縮文書が図24（C）に示すようなパターン⑤の圧縮XML文書である場合、その文書には、“<!DOCTYPE”は記述されているが、その後には“[”が記述されることなく、外部ファイル100におけるDTDを指定するシステム識別子が記述

されているので、処理はステップS134のNOルートからステップS139へ移行し、図24 (G) に示す新旧DTD表を入力してから、その新旧DTD表に従って、文書実現値における文書型宣言のシステム識別子“SYSTEM”により指定されるファイル名を、「..\\book2.dtd」から元のファイル名「..\\book.dtd」に書き換える（ステップS140）。

【0226】そして、図24 (E) や図24 (F) に示すタグ辞書90を用いて、前記文書実現値文字列逆置換手段により、文書実現値のタグ内に記述された1バイトの短縮文字列「a」、「b」、「c」や「A」が、それぞれ元の要素名「book」、「title」、「author」や属性名「field」に逆置換される。これにより、例えば図24 (C) に示す圧縮XML文書は、図24 (A) に示すようなXML文書に逆置換・伸長され、伸長文書として文書記憶部10等へ出力・格納される（ステップS142）。

【0227】このとき、ファイル名「..\\book.dtd」のDTD、つまり図24 (B) に示すDTDは外部ファイル100に保存されているので、図24 (D) に示す圧縮DTDを前記DTD文字列逆置換手段により逆置換・伸長して、図24 (B) に示すDTDを得る必要はない。

【0228】このように、本発明の第2実施形態によれば、文書実現値のタグ内やDTDの記述を解析し、その解析結果に従ってタグ辞書90を作成し、そのタグ辞書90を用いて、文書実現値のタグ内やDTDに記述された文字列を短縮文字列（1または2バイトの置換文字）に置き換えることにより、XML文書の特徴や構造を損なうことなくタグ内やDTDの文字列が圧縮されるので、XML文書の圧縮率を大幅に高めることができ、ひいては、大規模なデータベースを取り扱うシステムにおいて文書データの格納効率を大幅に高めることができる。

【0229】このとき、タグ内やDTDに記述された要素名および属性名を短縮文字列に置き換えることにより、検索可能な状態に保持したままタグ部分やDTDを圧縮することができる。つまり、要素名および属性名について置換を行なうとともにタグ辞書90を保持し、属性値は元の形のまま保持することで、文書データを伸長することなく圧縮した状態のままで検索や文書構造の把握を行なえるようになっている。

【0230】従って、圧縮後にXML文書の構造を解析して検索を行なう際に、圧縮されたXML文書の伸長を行なう必要がなく、大規模なデータベースにおいて文書データを圧縮格納しても、その文書データの検索処理等を短時間で行なうことができる。

【0231】属性名と属性値とはそれぞれ検索の対象となり得るため、属性名と属性値との両方を一体的に圧縮している場合、伸長・復元を行なわない限り検索を行な

うことができなくなる。そこで、第2実施形態では、属性名のみを短縮文字列に置換して圧縮を行ない、属性値は元の形のままとしておくことにより、ブラウザなどの応用ソフトウェアにおいては、圧縮文書の伸長処理を行なうことなく、タグ辞書90を参照しながら圧縮文書内のデータの検索を行なうことができる。

【0232】〔3〕第3実施形態の説明

次に、本発明の第3実施形態について説明する。本発明の第3実施形態では、第2実施形態と同様、図30 (A) に示すごとく、圧縮前のXML文書において、「<」と「>」とで囲まれた領域（DTDの宣言文や、文書実現値における開始タグおよび終了タグ）に記述された文字列を、図30 (C) に示すようなタグ辞書を使用して、短縮文字列（置換文字）に変換して圧縮する。このような圧縮を行なった場合、圧縮後のXML文書には、「<」および「>」を用いた記述形式は保存されたままであり、圧縮後も、XML文書としての特徴や構造は損なわれることはない。

【0233】また、第3実施形態では、平文の記述に用いられる言語の種類を識別した上で、該当する言語の辞書（単語辞書）、例えば図30 (D) に示すような日本語辞書を選択し、平文を成す単語を、最長一致法（longest match）で固定バイト長の短縮文字列（単語番号）に変換する。ここで、平文とは、開始タグと終了タグとの間に記述された内容のことという。

【0234】このようにして、例えば図30 (A) に示すような圧縮前のXML文書を、図30 (B) に示すようなXML文書に変換・圧縮することができる。なお、第3、4実施形態では、言語の種別は、例えば日本語、英語、中国語の3種類の中から識別される。

【0235】また、DTDをもたない整形式XML文書（パターン①）の場合は、文書実現値におけるタグ内の記述を調べてから、タグ辞書を用いて、タグ内における要素名等の文書文字列と短縮文字列（置換文字）との対応付けを行なった上で、XML仕様では言語識別用の属性（xml : lang）を見て平文（内容）の言語を識別してその言語に応じた辞書（単語辞書）を選択し、上述と同様の圧縮を行なう。第3実施形態の圧縮手法を、DTDをもたない整形式XML文書に適用した場合の具体例については、図32 (A) ～図32 (D) を参照しながら後述する。

【0236】以下、図25～図33を参照しながら、本発明の第3実施形態について説明する。まず、図30～図33により圧縮前のXML文書が圧縮後にどのようになるのかについて説明する。

【0237】図30 (A) ～図30 (D) はいずれも第3実施形態によるXML文書（パターン④：DTDを内部に記述した検証済みXML文書）の具体的な圧縮処理を説明するための図で、図30 (A) は内部記述したXML文書の圧縮前記述例を示し、図30 (B)

はその圧縮後記述例を示し、図30（C）はその圧縮処理に使用したタグ辞書の登録内容例を示し、図30

（D）はその圧縮処理に使用した日本語辞書の登録内容例を示している。

【0238】図31（A）～図31（G）はいずれも第3実施形態によるXML文書（パターン⑤：別ファイルのDTDを指定して利用するXML文書）の具体的な圧縮処理を説明するための図で、図31（A）は別ファイルに格納されたDTDを参照・利用するXML文書の圧縮前記述例を示し、図31（B）はその別ファイルに格納されたDTDの圧縮後記述例を示し、図31（C）はそのXML文書の圧縮後記述例を示し、図31（D）はそのDTDの圧縮後記述例（新規ファイルのDTD）を示し、図31（E）はその圧縮処理に使用したタグ辞書の登録内容例を示し、図31（F）はその圧縮処理に使用した日本語辞書の登録内容例を示し、図31（G）は新旧DTDの対応関係を保持する対応表の登録内容例を示している。

【0239】図32（A）～図32（D）はいずれも第3実施形態によるXML文書（パターン⑥：DTDをもたない整形式XML文書）の具体的な圧縮処理を説明するための図で、図32（A）はDTDをもたない整形式XML文書の圧縮前記述例を示し、図32（B）はその圧縮後記述例を示し、図32（C）はその圧縮処理に使用したタグ辞書の登録内容例を示し、図32（D）はその圧縮処理に使用した日本語辞書の登録内容例を示している。

【0240】図33は第3実施形態でのXML文書の圧縮手法を説明するための図で、この図33は、XML文書において“<”と“>”とで囲まれた箇所に記述された文字列中に、空白（スペース）が存在する場合の、置換文字への変換手法を説明するためのものである。

【0241】まず、図30（A）～図30（D）により、DTDを内蔵記述するXML文書に対する圧縮処理について説明する。図30（A）に示すごとく、圧縮前のXML文書には、1行目に示すXML宣言、2～7行目に示すDTD、8～12行目に示す平文の内容等が種々の記号とともに記入されている。

【0242】図30（A）において、1行目には、この文書がバージョン1.0のXML文書であることを示すXML宣言が記述され、2行目には、この文書の文書型名（DOCTYPEの名前）が「book」であることが記述され、その直後の「[」と7行目の「]」との間の記述が、この文書の構成を定義するものである。

【0243】また、3行目の最上位要素名の名前「book」と文書型名「book」とは一致することが必要である。そして、3行目には、要素「book」が子要素「chapter」を有して構成されることが内容モデルとして記述され、4～6行目には、要素「chapter」が、さらに、2つの子要素「title」および「paragraph」で構成され

ることが記述され、7行目の「]」により、2行目から始まった文書型宣言（DOCTYPE宣言、DTD記述）の内部サブセット記述が終了することが示されている。

【0244】終く8行目の“<book>”は要素「book」の開始タグであり、12行目の“</book>”は要素「book」の終了タグである。そして、これらのタグ間（9～12行目）の記述が、要素「book」の内容であり、9行目の記述は、要素「chapter」の子要素「title」の内容が「XMLの概要」であることを示し、10行目における、要素「chapter」の子要素「paragraph」の内容が「XMLとは…」であることを示す。また、9行目において、“<title>”は要素「title」の始まりを示す開始タグであり、“</title>”は要素「title」の終了を示す終了タグである。11行目の“</chapter>”は要素「chapter」の終了を示す終了タグである。

【0245】図30（A）に示す圧縮前のXML文書における文字列を、図30（C）に示すタグ辞書と、図30（D）に示す日本語辞書とを使用して、置換文字や單語番号に変換することにより、そのXML文書を、図30（B）に示すように圧縮する。ここで、図30（C）に示すタグ辞書は、図29を参照しながら後述する手法により作成され、また、図30（D）に示す日本語辞書は、予め作成された詩的な辞書である。

【0246】図30（A）の例では、上述したタグ辞書により、“chapter”，“title”，“paragraph”が、それぞれ固定長の置換文字“b”，“c”，“d”に置換・圧縮され、また、上述した日本語辞書により、平文の部分の「XML」，「の」，「概要」，「とは」が、それぞれ固定長の單語番号“α”，“β”，“γ”，“δ”等に置換・圧縮され、図30（B）に示すような圧縮後のXML文書が得られる。

【0247】次に、図31（A）～図31（G）により、XML文書とは別のファイルに格納されたDTDを参照する場合の圧縮手法について説明する。図31（A）に示す圧縮前のXML文書では、2行目の下線を付した部分、つまりシステム識別子「SYSTEM ".. /book.dtd"」により、ファイル名「book.dtd」をもつ別ファイルのDTDが指定されている。この別ファイルのDTDは、図31（B）に示すように記述されている。

【0248】図31（A）に示す圧縮前のXML文書における文字列を、図31（E）に示すタグ辞書と、図31（F）に示す日本語辞書とを使用して、置換文字や單語番号に変換することにより、そのXML文書を、図31（C）に示すように圧縮する。

【0249】また、図31（B）に示すDTDにおける文字列を、図31（E）に示すタグ辞書を使用して置換文字に変換することにより、そのDTDを、図31（D）に示すような新規ファイルのDTDとして圧縮・

作成する。この新規ファイルのファイル名としては、例えば “book2.dtd” が付与される。

【0250】なお、図31(C)に示すように、圧縮後のXML文書の2行目には、新規ファイルのDTDを指定すべく、そのファイル名“book2.dtd”が、旧ファイルのファイル名“book.dtd”に代えて記入される。そして、これら新規のDTDのファイル名と圧縮前の旧DTDのファイル名とが、図31(G)に示すごとく、新旧DTD対応表に記入される。ここで、図31(E)に示すタグ辞書も、図30(C)に示すタグ辞書と同様、図29を参照しながら後述する手法により作成され、また、図31(F)に示す日本語辞書は、予め作成された頃約束辞書である。

【0251】図31（A）の例でも、上述したタグ辞書により、「chapter」、「title」、「paragraph」が、それぞれ固定長の置換文字「b」、「c」、「d」に置換・圧縮され、また、上述した日本語辞書により、平文の部分の「XML」、「の」、「概要」、「とば」が、それぞれ固定長の単語番号「 α 」、「 β 」、「 γ 」、「 δ 」等に置換・圧縮され、図31（C）に示すような圧縮後のXML文が得られる。

【0252】また、図31（C）に示すDTDは、上述したタグ辞書により、“chapter”，“title”，“paragraph”が、それぞれ固定長の置換文字“b”，“c”，“d”に置換・圧縮され、ファイル名“book2.dtd”的新規ファイルに、新たなDTDとして格納される。

【0253】次に、図32(A)～図32(D)により、DTDをもたない整形式XML文書に対する圧縮処理について説明する。図32(A)に示す圧縮前のXML文書では、1行目のXML宣言についてDOCTYP E宣言で開始されるDTD文書型定義が述されていない。このような場合も、図32(A)に示す圧縮前のXML文書における文字列は、図32(C)に示すタグ辞書と図32(D)に示す日本語辞書を使用して、置換文字や単語番号に変換され、そのXML文書は、図32(B)に示すように圧縮される。

【0254】ここで、図32（C）に示すタグ辞書も、図30（C）や図31（E）に示すタグ辞書と同様、図29を参照しながら後述する手法により作成され、また、図32（D）に示す日本語辞書は、予め作成された静的な辞書である。図32（A）の例でも、上述したタグ辞書により、“charter”, “title”, “paragraph”など、文字や記号を意味するタグが用いられる。

“が”、それと固定長の直接文子 β , γ , δ ,
 “d”に置換・圧縮され、また、上述した日本語辞書により、
 文語の部分の“XML” , “の” , “概要” ,
 “とは”が、それぞれ固定長の単語番号 “ α ” ,
 “ β ” , “ γ ” , “ δ ”等に置換・圧縮され、図3-2

【0255】また、本実験形態では、図3.3に示すよう

に、XML文書において“<”と“>”とで囲まれた領域の文字列中に、空白（スペース）が存在する場合、その文字列を空白部分で区切る。このようにして区切って得られた文字列部分のそれぞれに、置換文字（短縮文字列）に対応させる。

【0256】即ち、図33に示す例では、まず、“<”と“>”と囲まれた「table orient="PORT" tocentry="1"」という開始タグが存在するが、これ全体を1つのタグとして登録せしめ、この文字列内に存在する空白部を自分で区切ることにより、この文字列を、「table」、「orient="PORT"」、「tocentry="1"」という3つの部分に区切る。

【0257】そして、区分けされた各部分に対して、図33に示すごとく置換文字（短縮文字列）“e”，“f”，“g”を付加する。このようにして登録内容を短くすることにより、タグ辞書のサイズを小型化することができ、検索範囲も小さくすることができ、ひいては検索手段のハード量も小さくすることができる。

【0258】なお、図3-3に示す例では、置換文字“e”、“f”、“g”に置き換えられた部分と、置換文字“h”および“j”に置き換えられた部分とがタグである。そして、置換文字“h”に置き換えられたタグがタイトルの始まりを示し、置換文字“j”に置き換えられたタグがタイトルの終わりを示している。また、置換文字“1”に置き換えられた部分が、タイトルの内容を示す平文“1”に置き換えられた文字である。また、置換文字“e”、“f”、“g”に置き換えられた部分は、それぞれ、<table orient="PORT" tocentry="1">というタグの要素名、第1属性（属性名=属性値）および第2属性（属性名=属性値）である。

【0259】次に、上述した圧縮処理に関する第3実施形態の圧縮装置および伸長装置の構成および動作について、図25～図29を参照しながら説明する。ここで、図25は本発明の第3実施形態としての構造化文書の圧縮装置の機能構成を示すブロック図、図26は本発明の第3実施形態としての構造化文書の伸長装置の機能構成を示すブロック図、図27は図25に示す圧縮装置によるタグ辞書作成手順（タグ辞書登録手順）を説明するためのフローチャート、図28は図25に示す圧縮装置による圧縮手順を説明するためのフローチャート、図29は図26に示す伸長装置による伸長手順を説明するためのフローチャートである。

【0260】ます、図25を参照しながら、第3実施形態の圧縮装置について説明すると、この図25において、101は文書記憶部、102はDTD条件検索部、103はタグ平文識別部、104はタグ文字列登録部、105はタグ辞書、106は文字列比較部、107は言語読取部、108は日本語辞書、109は中国語辞書、110は英語辞書、111はタグ文字列変換部、112は規則式文書検索部、113は単語系検索部、114は

はDTD記入部である。

【0261】文書記憶部101は、圧縮すべきXML文書を入力保持するもので、例えば図30（A）に示すような圧縮前のXML文書が保持されるメモリである。DTD条件調査部102は、圧縮すべきXML文書が、図30（A）に示すごとくDTDを内蔵しているか、図31（A）に示すごとく別ファイルのDTDを参照するものか、図32（A）に示すごとくDTDなしのものかの3通りのいずれであるかを識別するもので、第1および第2実施形態のパターン認識機能を実現するものである。

【0262】より具体的に、DTD条件調査部102は、圧縮対象のXML文書の2行目に“`<!DOCTYPE`”が記述されていればDTDを有するものと識別し、文書型名（ここでは“book”）つまり文書宣言の後に“`[`”が記述されればDTDを内蔵したXML文書であると識別し、“`[`”が記述されていなければ外部ファイルのDTDを参照するXML文書であると識別する。そして、XML文書の2行目に“`<!DOCTYPE`”が記述されていなければ、DTDなしのXML文書であると識別する。

【0263】タグ平文識別部103は、XML文書の注目文字列がタグか平文かを識別するものであり、その注目文字列が“`<`”および“`>`”により前後から囲まれていれば、その注目文字列をタグ内の記述であると識別する一方、囲まれていなければ、その注目文字列を平文内の記述であると識別する。例えば、図30（A）の9行目の“`XMLの概要`”や10行目の“`XMLとは…`”、あるいは、図30（B）の9行目の“`αβγ`”や10行目の“`αδ…`”は、“`<`”および“`>`”により囲まれていないので、平文であると識別される。

【0264】タグ文字列登録部104は、例えば図30（C）に示すようなタグ辞書105を登録・作成するものであり、そのタグ辞書105は、タグ内において例えば図33に示すごとく区分された文字列（すなはち「要素名」や「属性名=“属性値”」等）を、それぞれ短縮文字列（置換文字）に対応させるためのものである。上述したタグ平文識別部103やタグ文字列登録部104は、第2実施形態における文書実現値解析部20、DTD解析部30やタグ辞書作成部80に対応した機能を果たすものである。

【0265】ここで、タグ文字列登録部104は、図27に示すフローチャート（ステップS241～S251）に従って、上述のようなタグ辞書105を作成するようになっており、図27を参照しながら、タグ文字列登録部104によるタグ辞書作成手順（タグ辞書登録手順）について説明する。

【0266】まず、タグ文字列登録部104に文字を順次入力し（ステップS241）、入力された文字がEOF（End Of File）か否かを判別し（ステップS24

2）、EOFであれば（YESルート）、タグ辞書作成動作を終了する。ステップS242において入力文字がEOFでないと判別した場合（NOルート）には、その入力文字が“`<`”か否かを判別し（ステップS243）、“`<`”でなければ（NOルート）、ステップS241に戻って次の文字を入力する。

【0267】ステップS243において入力文字が“`<`”であると判別した場合（YESルート）には、メモリの文字列を空（#）にしてから（ステップS244）、“`<`”以降の文字（タグ内の文字）を順次入力する（ステップS245）。なお、ステップS241によって入力されてからステップS244でメモリから消された文字列は、“`<`”の前に記述された平文である。

【0268】ステップS245で入力された文字が空白か否かを判別し（ステップS246）、空白である場合（YESルート）、その空白が認識されるまでにメモリに蓄積された文字列に対して適当な置換文字（例えば“`b`”）を決め、その置換文字と文字列とを対応させてタグ辞書105に登録する（ステップS247）。この後、ステップS244に戻り、同様の処理を繰り返し実行する。

【0269】ステップS246において空白ではないと判別した場合（NOルート）には、ステップS245で入力された文字が“`!`”か否か判別し（ステップS248）、“`!`”であれば（YESルート）、この“`!`”に続く文字列はタグではなくコメントなので、ステップS241に戻る。

【0270】ステップS248において“`!`”ではないと判別した場合（NOルート）には、ステップS245で入力された文字が“`>`”か否かを判別し（ステップS249）、“`>`”であれば（YESルート）、その“`>`”が認識されるまでにメモリに蓄積された文字列に対して適当な置換文字を決め、その置換文字と文字列とを対応させてタグ辞書105に登録する（ステップS250）。

【0271】そして、ステップS249において“`>`”を認識したということは、タグの記述が終わったことを意味するので、ステップS241に戻り、同様の処理を繰り返し実行する。また、ステップS249において“`>`”ではないと判別した場合（NOルート）には、今回、ステップS245で入力された文字を、メモリに蓄積されている文字列に加えて新文字列としてから（ステップS251）、ステップS245に戻る。

【0272】従って、図33に示すような文字列が入力された場合、空白毎に文字列が区切られ、区切られた文字列「table」、「orient="PORT"」、「tocentry="1"」に対しそれぞれ置換文字e, f, gが決められ、これらの文字列と置換文字との対応関係が、順々、タグ辞書105に登録される。また、図30（A）の9行目に記述された文字列が入力された場合、“`<`”および“`>`”が

認識される都度、その文字列が区切られ、区切られた文字列「chapter」、「title」に対しそれぞれ置換文字

b, cが決められ、これらの文字列と置換文字との対応関係が、順次、タグ辞書105に登録される。

【0273】さて、図25に示す圧縮装置において、タグ辞書105は、タグ文字列登録部104により前述のごとく登録・作成されたもので、例えば図30(C), 図31(E)や図32(C)に示すような、置換文字とタグ内の文字列との対照表であり、メモリに保持される。

【0274】文字列比較部106は、文書記憶部101に保持された文字列と、辞書105, 108~110の登録文字列または単語文字列とを比較し、その文字列が登録文字列または単語文字列と一致した場合には、その文字列に対して置換文字を出力するものである。このとき、タグ内の文字列はタグ辞書105の登録文字列と比較され、平文部分の文字列は、後述する単語辞書108~110のいずれかにおける単語文字列と比較される。文字列を、文字列比較部106によって出力された置換文字(短縮文字列)へ変換することにより、XML文書が圧縮されることになる。

【0275】言語識別部107は、圧縮対象のXML文書の内容(平文)において記述される言語が何語であるかを、XML宣言におけるエンコーディング宣言(図示省略)に記述された文字コード名、並びに、任意のタグにおける言語識別用の属性(xml: lang)の値を解読することによって識別するものである。そして、本実施例の言語識別部107は、その言語が例えば日本語、中国語、英語の3つのうちのいずれであるかを識別し、その識別結果に対応して、日本語辞書108、中国語辞書109、英語辞書110のうちのいずれか一つを平文用の単語辞書として選択するものである。

【0276】日本語辞書108は、XML文書の平文において記述される言語が日本語の場合に、例えば図30(D)に示すように、平文を構成する日本語の単語文字列とそれに対応する単語番号(短縮文字列、置換文字)との対応をとるための対照表であり、事前に構成された既知のものである。ここで、単語番号は、平文内の実際の単語文字列よりも短く、且つ、その単語文字列を特定しうる、固定バイト長の短縮文字列(置換文字)である。同様に、中国語辞書109や英語辞書110のいずれも、平文を構成する各国語の単語文字列と単語番号(短縮文字列、置換文字)との対応をとるための対照表である。

【0277】タグ文字列変換部111は、タグの文字列を、文字列比較部106から的一致信号に応じて、この文字列比較部106で付与された置換文字(短縮文字列)に変換する、変換処理を行なうものである。そして、このタグ文字列変換部111と上述した文字列比較部106とが、第2実施形態の文書実現値文字列置換部

41やDTD文字列置換部51に対応した機能を果たすものである。

【0278】同様に、単語番号変換部112は、平文の文字列を、文字列比較部106からの一致信号に応じて、この文字列比較部106で付与された単語番号(置換文字、短縮文字列)に変換するものであり、最長一致法により単語を固定バイトの単語番号に変換するものである。

【0279】単語番号ファイル113は、タグ文字列変換部111からの置換文字変換出力と、単語番号変換部112からの単語番号変換出力により得られた、例えば図30(B)に示すようなXML文書の圧縮データを保持するものである。DTD記入部114は、図31(A)に示すごとき、別ファイルのDTDを参照するXML文書に対して、新規ファイル名を記入するものである。

【0280】次に、図26を参照しながら、第3実施形態の伸長基準について説明する。なお、図中、既述の符号と同一の符号は同一もしくはほぼ同一の部分を示しているので、その説明は省略する。図26において、120は単語番号ファイル記憶部、121は置換文字比較部、122はタグ文字列逆変換部、123は単語番号逆変換部、124は文書記憶部、125は旧DTD記入部である。

【0281】単語番号ファイル記憶部120は、伸長対象である、例えば図30(C)に示すような圧縮後のXML文書を保持するものである。置換文字比較部121は、単語番号ファイル記憶部120に保持された置換文字(単語番号)と、辞書105、108~110の置換文字(単語番号)とを比較し、これらの置換文字が一致した場合には、その置換文字に対して文字列(単語文字列)を出力するものである。このとき、タグ内の置換文字はタグ辞書105に登録された置換文字と比較され、平文部分の単語番号(置換文字)は、単語辞書108~110のいずれかにおける単語文字列と比較される。

【0282】タグ文字列逆変換部122は、圧縮後のXML文書における置換文字を、置換文字比較部121からの信号に基づき、その置換文字に対応した文字列に逆変換するものである。同様に、単語番号逆変換部123は、圧縮後のXML文書における単語番号(置換文字)を、番号比較部121からの信号に基づき、その単語番号に対応した単語文字列に逆変換するものである。

【0283】文書記憶部124は、タグ文字列逆変換部122から出力された、置換文字に対応したタグ文字列と、単語番号逆変換部123から出力された、単語番号に対応した単語文字列を記述されたXML文書(復元された文書)を保持するものである。旧DTD記入部125は、XML文書において、前記圧縮時に記入された新DTDを旧DTDに復元するものである。

【0284】次に、図25にて説明した第3実施形態の

圧縮装置の動作について、図30～図32を参照しながら、図28に示すフローチャート（ステップS201～S217）に従って説明する。まず、文書記憶部101に、例えばオペレータが作成済みの図30（A）に示すようなXML文書を保持させる。DTD条件調査部102は、文書記憶部101に格納された、圧縮対象のXML文書を参照して、そのXML文書中に“<!DOCTYPE”が記述されているか否かをチェックし（ステップS201）、“<!DOCTYPE”が記述されている場合（YESルート）には、そのXML文書はDTDを有するものと判別される。なお、DTDがないと判別された場合（NOルート）、ステップS203へ移行する。

【0285】続いて、DTD条件調査部102は、DTDを有するXML文書に“[”が記述されているか否かを判別する（ステップS202）。そして、“[”が記述されている場合（YESルート）、DTDがXML文書中に内蔵されているものと判別してステップS203へ移行する一方、“[”が記述されていない場合（NOルート）、そのXML文書についてのDTDは外部ファイルに格納されているものと判別してステップS209へ移行する。

【0286】DTDを内蔵したXML文書に対しては、以下のステップS203～S208の処理を施して圧縮を行なう。まず、タグ文字列登録部104により、図27に示すフローチャートに従ってタグ文字列の置換文字登録（タグ辞書作成）処理を行ない、図30（C）に示すようなタグ辞書105を作成する（ステップS203）。

【0287】続いて、タグ平文識別部103が、文書記憶部101に保持されたXML文書から平文を識別して言語識別部107に送出する。このとき、言語識別部107は、タグ平文識別部103からのXML文書のエンコーディング宣言において記述された文字コード名（図示省略）、並びに、任意のタグにおける言語識別用の属性（xal: lang）の値を解説することにより、平文の言語が何語であるかを識別し、その言語に応じた単語辞書、例えば日本語の場合は日本語辞書108を選択する（ステップS204）。

【0288】そして、文字列比較部106は、タグ辞書105の登録文字列と、圧縮対象のXML文書におけるタグ内の文字列とを比較し、これらの文字列が一致した場合、その登録文字列に対応した置換文字（短縮文字列）をタグ辞書105から読み出し、タグ文字列変換部111にて、タグ内の文字列を置換文字に変換する（ステップS205）。

【0289】ついで、文字列比較部106は、図30（D）に示すような日本語辞書108の登録単語文字列と、圧縮対象のXML文書における平文内の単語文字列とを比較し、これらの単語文字列が一致した場合、その

登録単語文字列に対応した単語番号（置換文字、短縮文字列）を日本語辞書108から読み出し、単語番号変換部112にて、平文内の単語文字列を単語番号に変換する（ステップS206）。

【0290】この後、タグ辞書105は、後述する逆変換処理つまり伸長・復元の際に必要になるため、図示省略のファイルにに出力される（ステップS207）。また、ステップS205において置換文字に変換されたタグと、ステップS206において単語番号に変換された平文とは、図30（B）に示すように圧縮されたXML文書と、図30（C）に示すように圧縮されたXML文書と、図30（D）に示すように圧縮されたXML文書と、図示省略の記憶装置等で保持される（ステップS208）。

【0291】一方、外部ファイルに格納されたDTDを用いるXML文書に対しては、以下のステップS209～S217の処理を施して圧縮を行なう。ステップS202において“[”が記述されていないと判別された場合、DTD条件調査部102は、圧縮対象のXML文書についてのDTDを外部参照により認識する必要があると判別し、例えば図31（A）の2次目に示すごくシステム識別子によって指定されたファイル名“book.dtd”に基づいて、別ファイルのDTD（例えば図31

（B）参照）が認識・参照される。そして、そのDTDに対してタグ文字列登録部104がタグ文字列登録処理を行ない、図31（E）に示すようなタグ辞書105が作成される（ステップS209）。

【0292】続いて、ステップS204と同様、タグ平文識別部103が、文書記憶部101に保持されたXML文書から平文を識別して言語識別部107に送出する。このとき、言語識別部107は、タグ平文識別部103からのXML文書のエンコーディング宣言において記述された文字コード名（図示省略）を解説することにより、平文で記述される言語が何語であるかを識別し、その言語に応じた単語辞書、例えば日本語の場合は日本語辞書108を選択する（ステップS210）。

【0293】また、DTD条件調査部102は、DTD記入部114を制御して、例えば図31（G）に示すごとく新旧DTD名に新規ファイルのDTD名「book2.dtd」を記入させるとともに、単語番号変換部112による処理に際して、圧縮前の旧DTD名「book.dtd」を新規ファイルのDTD名「book2.dtd」に書換・記入させる（ステップS211）。

【0294】そして、文字列比較部106は、タグ辞書105の登録文字列と、図31（B）に示すような別ファイルのDTDにおけるタグ内の文字列とを比較し、これらの文字列が一致した場合、その登録文字列に対応した置換文字（短縮文字列）をタグ辞書105から読み出し、タグ文字列変換部111にてタグ内の文字列を置換文字に変換することにより、図31（D）に示すような新規ファイルのDTDを圧縮作成する（ステップS21

2)。

【0295】また、ステップS205と同様、文字列比較部106は、タグ辞書105の登録文字列と、文書記憶部101に格納された圧縮対象のXML文書におけるタグ内の文字列とを比較し、これらの文字列が一致した場合、その登録文字列に対応した置換文字（短縮文字列）をタグ辞書105から読み出し、タグ文字列変換部111にて、タグ内の文字列を置換文字に変換する。そして、タグ文字列変換部111は、上述のごとくタグ内の文字列を置換文字に変換・圧縮したXML文書を単語番号変換部112に送出する（ステップS212）。

【0296】ついで、文字列比較部106は、図31(F)に示すような日本語辞書108の登録単語文字列と、タグ文字列変換部111からのXML文書（タグ内の文字列が置換文字に変換された文書）における平文内の単語文字列とを比較し、これらの単語文字列が一致した場合、その登録単語文字列に対応した単語番号（置換文字、短縮文字列）を日本語辞書108から読み出し、単語番号変換部112にて平文内の単語文字列を単語番号に変換する。このとき、DTD記入部114により、XML文書において旧DTD名「book.dtd」が新しいDTD名「book2.dtd」に書き換えられる。このようにして、図31(A)に示すような圧縮前のXML文書は図31(C)に示すようなXML文書に圧縮される（ステップS213）。

【0297】この後、タグ辞書105は、ステップS207と同様、後述する逆変換処理つまり伸長・復元の際に必要になるため、図示省略のファイルに出力される（ステップS214）。また、ステップS213において変換された、図31(C)に示すような圧縮後のXML文書は、単語番号変換部112から単語番号ファイル113に出力される（ステップS215）。

【0298】さらに、ステップS212で圧縮作成された新規ファイルDTDは図示省略のファイルに出力されるとともに（ステップS216）、ステップS211で作成した、図31(G)に示すような新旧DTD対応表も、後述する逆変換処理つまり伸長・復元の際に必要になるため、図示省略のファイルに出力される。なお、DTDをもたないXML文書に対しても、前述したステップS203～S208の処理が施され、例えば図32(A)に示すXML文書が図32(B)に示すごとく圧縮される。

【0299】次に、図26にて説明した第3実施態形の伸長装置の動作について、図30～図32を参照しながら、図29に示すフローチャート（ステップS221～S232）に従って説明する。まず、図示省略のファイルからタグ辞書105を取り出し、そのタグ辞書105を図示省略のメモリに格納する（ステップS221）。

【0300】そして、そのタグ辞書105に対応する、圧縮XML文書の単語番号ファイル113を記憶部12

0に格納する（ステップS222）。ステップS221およびS222の処理により、図30(C)に示すタグ辞書に対しては図30(B)に示すごとく圧縮されたXML文書が記憶部120に格納され、図31(E)に示すタグ辞書に対しては図31(C)に示すごとく圧縮されたXML文書が記憶部120に格納され、図32(C)に示すタグ辞書に対しては図32(B)に示すごとく圧縮されたXML文書が記憶部120に格納される。【0301】この後、DTD条件調査部102は、記憶部120から伸長復元対象のXML文書を読み出し、まず、そのXML文書に“<!DOCTYPE”が記述されているか否かをチェックする（ステップS223）。“<!DOCTYPE”が記述されている場合（YESルート）、DTD条件調査部102は、さらに “[” が記述されているか否かをチェックし（ステップS224）、 “[” が記述されている場合（YESルート）、伸長復元対象のXML文書にはDTDが内蔵されているものと判断する。

【0302】このように伸長復元対象のXML文書がDTDを内蔵している場合、言語識別部107は、任意のタグ中の言語識別用の属性（xml:lang）の値からそのXML文書の平文の言語を識別し、予め作成されている単語辞書部の中から、その識別結果に応じた単語辞書、例えば図31(D)に示すような日本語辞書108を選択する（ステップS225）。

【0303】そして、タグ平文識別部103は、圧縮されたXML文書のタグ部分を識別し、タグ辞書105の置換文字と記憶部120から抽出したタグ部分の置換文字とを置換文字比較部121で比較・照合させ、これらの置換文字が一致した場合、その置換文字に対応するタグ内文字列をタグ辞書105から読み出してタグ文字列逆変換部122に送出し、このタグ文字列変換部122において、圧縮されたXML文書の置換文字をタグ内文字列に変換する、タグ文字列逆変換処理が行なわれる（ステップS226）。これにより、例えば図30(B)の置換文字b, c, dが、図30(A)に示すごとく、それぞれ“chapter”, “title”, “paragraph”に変換される。

【0304】ついで、タグ平文識別部103は、圧縮されたXML文書の平文部分を識別し、その平文部分の単語番号（置換文字）を、置換文字比較部121において、日本語辞書108の登録単語番号（登録置換番号）と比較し、これらの単語番号が一致した場合、その単語番号に対応する単語文字列を日本語辞書108から読み出して単語番号逆変換部123に送出し、この単語番号逆変換部123において、圧縮されたXML文書の単語番号を単語文字列に変換する、単語番号逆変換処理が行なわれる（ステップS227）。これにより、例えば図30(B)の単語番号α, β, γ, δが、図30(A)に示すごとく、それぞれ“XML”, “の”, “概

要”，“とは”に変換される。

【0305】ステップS226およびS227でそれ逆変換されたタグ文字列や平文は、例えば図30(A)に示すような圧縮前のXML文書、即ち復元文書となり、文書記憶部124に保持される。一方、ステップS224で伸長復元対象のXML文書に“[”が記述されていないと判別された場合(NOルート)、DTD条件調査部102は、そのXML文書についてのDTDは外部ファイルに格納しているものと判別し、そのXML文書の圧縮処理時に作成・保存された、例えば図32(G)に示すような新旧DTD対応表を入力する(ステップS228)。

【0306】また、言語識別部107は、ステップS225と同様、タグ中の言語識別用の属性(xml:lang)の値からそのXML文書の平文の言語を識別し、予め作成されている単語辞書群の中から、その識別結果に応じた単語辞書、例えば図32(F)に示すような日本語辞書108を選択する(ステップS229)。

【0307】さらに、DTD条件調査部102は、旧DTD記入部125を制御して、S228で読み出した新旧DTD対応表から、DTDの元のファイル名が“book.dtd”であることを認識するとともに、例えば図31(C)に示すような伸長復元対象のXML文書において、“book2.dtd”と記述されているDTD名を旧DTD名“book.dtd”に書換・記入させる(ステップS230)。

【0308】そして、ステップS226と同様に、タグ平文識別部103は、圧縮されたXML文書のタグ部分を識別し、タグ辞書105の置換文字と記憶部120から出力したタグ部分の置換文字とを置換文字比較部121で比較・照合させ、これらの置換文字が一致した場合、その置換文字に対応するタグ内文字列をタグ辞書105から読み出してタグ文字列逆変換部122に送出し、このタグ文字列逆変換部122において、圧縮されたXML文書の置換文字をタグ内文字列に変換する、タグ文字列逆変換処理が行なわれる(ステップS231)。

【0309】これにより、例えば図31(C)の置換文字b, c, dが、図31(A)に示すごとく、それぞれ“chapter”, “title”, “paragraph”に変換され、その変換結果は、単語番号変換部123に送出される。なお、このとき、ステップS230の処理により、XML文書中に記述されたDTD名は、ファイル名“book2.dtd”から旧ファイル名“book.dtd”に変換されている。

【0310】ついで、ステップS227と同様に、タグ平文識別部103は、圧縮されたXML文書の平文部分を識別し、その平文部分の単語番号(置換文字)を、置換文字比較部121において、日本語辞書108の登録単語番号(登録置換番号)と比較し、これらの単語番号が一致した場合、その単語番号に対応する単語文字列を

日本語辞書108から読み出して単語番号逆変換部123に送出し、単語番号逆変換部123において、圧縮されたXML文書の単語番号を単語文字列に変換する、単語番号逆変換処理が行なわれる(ステップS232)。

【0311】これにより、例えば図31(C)の単語番号a, b, r, 8が、図31(A)に示すごとく、それぞれ“XML”, “の”, “概要”, “とは”に変換される。このようにして逆変換されたタグ文字列や平文は、例えば図31(A)に示すような圧縮前のXML文書、即ち復元文書となり、文書記憶部124に保持される。

【0312】なお、ステップS223において伸長復元対象のXML文書に“<!DOCTYPE”が記述されていないと判別された場合(NOルート)、そのXML文書はDTDをもたないものと判別され、ステップS225へ移行し、そのXML文書に対しても、前述したステップS223～S227の処理が施され、例えば図32(A)に示すXML文書が図32(B)に示すごく伸長・復元されて文書記憶部124に保持される。

【0313】このように、上述した本発明の第3実施形態では、つまり文字列の置換文字(類似文字列)への変換処理を行なうことによりXML文書の圧縮を行ない、その際に用いられたタグ辞書105や各国語毎の単語辞書108～110は、いずれも圧縮されることなく保存されるので、XML文書を、伸長することなく圧縮した状態のまま検索することができ、例えば検索時間が0.1秒を超えてはいけない場合(システム)に用いて好適である。

【0314】次に、本発明の第3実施形態の変形例としての圧縮装置および伸長装置の構成および動作について、図34～図37を参照しながら説明する。ここで、図34は本発明の第3実施形態の変形例としての構造化文書の圧縮装置の機能構成を示すブロック図、図35は本発明の第3実施形態の変形例としての構造化文書の伸長装置の機能構成を示すブロック図、図36は第3実施形態の変形例における構造化文書の圧縮手順を説明するためのフローチャート、図37は第4実施形態の変形例における構造化文書の伸長手順を説明するためのフローチャートである。

【0315】まず、図34を参照しながら、第3実施形態の変形例としての圧縮装置について説明すると、この図34において、131は文書記憶部、132はタグ平文識別部、133はタグ文字列登録部、134はタグ辞書、135は文字列比較部、136は言語識別部、137は日本語辞書、138は中国語辞書、139は英語辞書、140はタグ文字列変換部、141は単語番号変換部、142は可変長符号化部、143は圧縮ファイル記憶部である。

【0316】文書記憶部131は、圧縮すべきXML文書を入力保持するもので、例えば図30(A)に示すような圧縮前のXML文書が保持されるメモリであり、図

25に示す文書記憶部101に対応するものである。タグ平文識別部132は、XML文書の注目文字列がタグが平文かを識別するものであり、図25におけるタグ平文識別部103に対応するものであり、このタグ平文識別部103と同様に動作する。

【0317】タグ平文列登録部133は、タグ内の文字列を置換文字(短縮文字列)に変換するためのタグ辞書134を作成するものであり、図25におけるタグ文字列登録部104に対応するもので、図27により前述した手順に従ってタグ辞書登録動作を行なう。タグ辞書134は、タグ平文列登録部133により作成されたもので、例えば図30(C)、図31(E)や図32(C)に示すような、置換文字とタグ内の文字列との対照表であり、メモリに保持される。このタグ辞書134は、図25におけるタグ辞書105に対応するものである。

【0318】文字列比較部135は、文書記憶部131に保持された文字列と、辞書134、137～139の登録文字列または単語文字列とを比較し、その文字列が登録文字列または単語文字列と一致した場合には、その文字列に対して置換文字を出力するものである。このとき、タグ内の文字列はタグ辞書134の登録文字列と比較され、平文部分の文字列は、後述する単語辞書137～139のいずれかにおける単語文字列と比較される。

文字列を、文字列比較部135から出力された置換文字(短縮文字列)へ変換することにより、XML文書が圧縮されることになる。この文字列比較部135は、図25における文字列比較部106に対応するものである。

【0319】言語識別部136は、圧縮対象のXML文書の内容(平文)において記述される言語が何語であるかを、XML宣言におけるエンコーディング宣言(図示省略)に記述された文字コード名、あるいは、任意のタグにおける言語識別用の属性(xml:lang)の値を解読することによって識別するものである。図34に示す言語識別部136も、その言語が例えば日本語、中国語、英語の3つのうちのいずれであるかを識別し、その識別結果に対応して、日本語辞書137、中国語辞書138、英語辞書139のうちのいずれか一つを平文用の単語辞書として選択するもので、図25における言語識別部107に対応するものである。

【0320】日本語辞書137は、XML文書の平文において記述される言語が日本語の場合に、例えば図30(D)に示すように、平文を構成する日本語の単語文字列とそれに対応する単語番号(短縮文字列、置換文字)との対応をとるための対照表であり、事前に構成された既知のもので、図25における日本語辞書108に対応するものである。

【0321】同様に、中国語辞書109や英語辞書110のいずれも、平文を構成する各国語の単語文字列と単語番号(短縮文字列、置換文字)との対応をとるための対照表であり、事前に構成された既知のもので、図25

における中国語辞書109や英語辞書110に対応するものである。タグ文字列変換部140は、タグの文字列を、文字列比較部135からの一致信号に応じて、この文字列比較部106で付与された置換文字(短縮文字列)に変換する、変換処理を行なうもので、図25におけるタグ文字列変換部111に対応するものである。

【0322】同様に、単語番号変換部141は、平文の文字列を、文字列比較部135からの一致信号に応じて、この文字列比較部106で付与された単語番号(置換文字、短縮文字列)に変換するものであり、最も一致法により単語を固定バイトの単語番号に変換するもので、図25における単語番号変換部112に対応するものである。

【0323】可変長符号化部142は、タグ文字列変換部140や単語番号変換部141による変換結果(タグ内やDTDに記述された文字列を置換文字に置き換え且つ平文内の単語文字列を単語番号に置き換えて得られた文字列)、つまり圧縮ファイルを、周知の手法で可変長符号化し、可変長符号化された圧縮ファイルを、タグ辞書とともにに出力するものである。圧縮ファイル記憶部143は、可変長符号化部142から出力された、タグ辞書と可変長符号化された圧縮ファイルとを保持するものである。

【0324】次に、図35を参照しながら、第3実施形態の変形例としての仲介装置について説明する。なお、図中、既述の符号と同一の符号は同一もしくはほぼ同一の部分を示しているので、その説明は省略する。図35において、151は圧縮ファイル記憶部、152は可変長復号化部、153は置換文字比較部、154はタグ文字列変換部、155は単語番号変換部、156は文書記憶部である。

【0325】圧縮ファイル記憶部151は、図34に示す圧縮装置により可変長符号化された圧縮ファイルを入力されるものであり、図34に示す圧縮ファイル記憶部143そのものであってもよく、あるいは、この圧縮ファイル記憶部143から出力された圧縮ファイルを保持するものとして構成してもよい。可変長復号化部152は、圧縮ファイル記憶部151から読み出された圧縮ファイルを、可変長符号化された状態から、例えば図30(B)に示すような圧縮ファイルに復号化するものである。

【0326】置換文字比較部153は、可変長復号化部152により復号化された圧縮ファイルの置換文字(単語番号)と、辞書134、137～139の置換文字(単語番号)とを比較し、これらの置換文字が一致した場合には、その置換文字に対して文字列(単語文字列)を出力するものである。このとき、タグ内の置換文字はタグ辞書134に登録された置換文字と比較され、平文部分の単語番号(置換文字)は、単語辞書137～139のいずれかにおける単語文字列と比較される。この置

換文字比較部153は、図26における置換文字比較部121に対応するものである。

【0327】タグ文字列逆変換部154は、可変長符号化部152により復号化された圧縮ファイル（圧縮後のXML文書）における置換文字を、置換文字比較部153からの信号に基づき、その置換文字に対応した文字列に逆変換するもので、図26におけるタグ文字列逆変換部122に対応するものである。同様に、単語番号逆変換部155は、可変長符号化部152により復号化された圧縮ファイル（圧縮後のXML文書）における単語番号を、置換文字比較部153からの信号に基づき、その単語番号に対応した単語文字列に逆変換するもので、図26における単語番号逆変換部123に対応するものである。

【0328】文書記憶部156は、タグ文字列逆変換部154から出力された、置換文字に対応したタグ文字列と、単語番号逆変換部155から出力された、単語番号に対応した単語文字列とを記述されたXML文書（復元された文書）を保持するもので、図26における文書記憶部124に対応するものである。

【0329】次に、図34にて説明した圧縮装置の動作について、図30を参照しながら図36に示すフローチャート（ステップS261～S267）に従って説明する。まず、文書記憶部101に、例えばオペレータが作成済みの図30（A）に示すようなXML文書を保持させる。そして、タグ平文識別部132は、文書記憶部131に保持されたXML文書からタグ部分を抽出し、そのタグ部分に基づいて、タグ文字列登録部133は、図27に示すフローチャートに従ってタグ文字列の置換文字登録（タグ辞書作成）処理を行ない、図30（C）に示すようなタグ辞書134を作成する（ステップS261）。

【0330】続いて、タグ平文識別部132が、文書記憶部131に保持されたXML文書から平文を識別して言語識別部136に送出する。このとき、言語識別部107は、タグ平文識別部132からのXML文書のエンコーディング宣言において記述された文字コード名（図示省略）、あるいは、任意のタグでの言語識別用の属性（xml:lang）の値を解読することにより、平文の言語が何語であるかを識別し、その言語に応じた単語辞書、例えば日本語の場合は日本語辞書137を選択する（ステップS262）。

【0331】そして、文字列比較部135は、タグ辞書134の登録文字列と、圧縮対象のXML文書におけるタグ内の文字列とを比較し、これらの文字列が一致した場合、その登録文字列に対応した置換文字（短縮文字列）をタグ辞書134から読み出し、タグ文字列逆変換部140にて、タグ内の文字列を置換文字に変換する（ステップS263）。

【0332】ついで、文字列比較部135は、図30

（D）に示すような日本語辞書137の登録単語文字列と、圧縮対象のXML文書における平文内の単語文字列とを比較し、これらの単語文字列が一致した場合、その登録単語文字列に対応した単語番号（置換文字、短縮文字列）を日本語辞書137から読み出し、単語番号変換部141において、平文内の単語文字列を単語番号に変換する（ステップS264）。

【0333】この後、タグ文字列逆変換部140や単語番号変換部141による変換結果（圧縮ファイル：図30（B）に示すようなXML文書）は、可変長符号化部142において、周知の手法で可変長符号化される（ステップS265）。また、タグ辞書134は、後述する逆変換処理つまり伸長・復元の際に必要になるため、図示省略のファイルに出力される（ステップS266）。さらに、可変長符号化部142は、S265にて可変長符号化した圧縮ファイルを、圧縮ファイル記憶部143にに出力する（ステップS267）。

【0334】次に、図35にて説明した伸長装置の動作について、図30を参照しながら図37に示すフローチャート（ステップS271～S276）に従って説明する。まず、図示省略のファイルからタグ辞書134を取り出し、そのタグ辞書134を図示省略のメモリに格納する（ステップS271）。そして、そのタグ辞書134における可変長符号化部142にてより符号化された圧縮ファイルを、圧縮ファイル記憶部51に入力・格納する（ステップS272）。

【0335】この後、可変長符号化部152により、圧縮ファイル記憶部151に入力された圧縮ファイルを復号して、例えば図30（B）に示すようなXML文書に復元する（ステップS273）。また、言語識別部136は、言語識別用タグ（図示省略）からそのXML文書の平文の言語を識別し、予め作成されている単語辞書群の中から、その識別結果に応じた単語辞書、例えば図30（D）に示すような日本語辞書137を選択する（ステップS274）。

【0336】そして、タグ平文識別部132は、圧縮されたXML文書の平文部分を識別し、その平文部分の単語番号（置換文字）を、置換文字比較部153において、日本語辞書137の登録単語番号（登録置換番号）と比較し、これらの単語番号が一致した場合、その単語番号に対応する単語文字列を日本語辞書137から読み出して単語番号逆変換部155に送出し、この単語番号逆変換部155にて、圧縮されたXML文書の単語番号を単語文字列に変換する（ステップS275）。

【0337】ついで、タグ平文識別部132は、圧縮されたXML文書のタグ部分を識別し、タグ辞書134の置換文字とタグ部分の置換文字を置換文字比較部153で比較・合算させ、これらの置換文字が一致した場合、その置換文字に応じたタグ内文字列をタグ辞書134から読み出してタグ文字列逆変換部154に送出

し、このタグ文字列変換部154において、圧縮されたXML文書の置換文字をタグ内文字列に変換する(ステップS276)。このようにして、図30(A)に示すようにXML文書が復元され、文書記憶部156に保持される。

【0338】本発明の第3実施形態によれば、圧縮率および検索速度は下記のようになる。

(i) 伸長せずに検索する方式つまり検索時間が0.1秒を超えてはいけない場合、タグの圧縮率および平文の圧縮率は下記のようになる。タグの圧縮率は、タグ内の文字列の長さをnバイトとすると、タグの元の長さは $(n+2)$ バイトであり、圧縮したタグの長さは $3/(n+2)$ バイトであり、圧縮率は $3/(n+2)$ となる。 $n=3$ とすると、0.6程度となる。また、平文の圧縮率は、大半の単語は4バイトから6バイトのため、0.3程度となる。検索速度としては、元の検索システムの検索速度(例えば、平均0.08秒)が保たれる。

【0339】(ii) 伸長して検索する方式つまり検索時間が0.1秒を超えてはよい場合、タグの圧縮率および平文の圧縮率は下記のようになる。タグの圧縮率は、タグ内の文字列の長さをnバイトとすると、タグの元の長さは $(n+2)$ バイトであり、圧縮したタグの長さは2バイト(固定バイト符号)であり、圧縮率は $2/(n+2)$ となる。 $n=3$ とすると、0.4程度となる。可変長符号化により0次文脈で圧縮しても圧縮率は同等である。また、平文の圧縮率は、静的辞書で単語番号変換し、0次文脈で圧縮した場合0.4程度である。検索速度は、検索システムの検索速度(例: 平均0.08秒)に対して0.02秒から0.03秒だけ速くなる。

【0340】ところで、特開平11-53349号公報に開示された技術では、タグ記号“<”および“>”と、これらに挟まれた文字列と一緒にして符号化しているので、第3実施形態のごく文字列を符号化するものとは異なり、図30(A)の3~6行目に示す“(chapter)”, “chapter”等は符号化することができないので、第3実施形態に比較して圧縮量が小さい。

【0341】本発明の第3実施形態によれば、下記のような利点が得られる。

A. タグ文字列登録部114や133によりタグ辞書105, 134を作成し、そのタグ辞書105, 134に基づいてタグ文字列を圧縮変換してタグを圧縮することにより、XML文書を圧縮するので、圧縮率を高めることができるものならず、このタグを圧縮した状態で検索可能にすることができる。

【0342】B. 図27や図33を参照しながら説明したごとく、タグ文字列をタグ辞書に登録する際に、“<”と“>”とに囲まれたタグ領域内(ただし“<”と“>”により囲まれたコメント領域を除く)において、空白文字が出現する度に文字列を区切って置換文字に対応させているので、タグ部分を正確に圧縮しな

がら圧縮率を高くすることができるほか、長いタグを短い文字で区切ることができ、タグ辞書のサイズを小形化し、タグ辞書を検索し易いものとして構成することができ、しかもタグを圧縮した状態で検索可能にことができる。

【0343】C. 長いタグ部分を短い文字で区切ることができ検索し易くすることができるのみならず、さらにこれを圧縮して、圧縮率をより高くすることができる。

D. XML文書中の文字コード名、あるいは、任意のタグでの言語識別別印の属性の値により平文の言語を認識し、その言語に応じた単語辞書を決定するので、例えば日本語、中国語、英語等の様々な言語により記述されたXML文書にも正確に対応することができる。

【0344】E. 平文の部分を各言語の単語辞書に登録された単語文字列と比較し、単語番号変換部112, 141において、登録単語文字列と一致した部分を固定長バイト番号(置換文字、短縮文字列)に変換するので、平文部分を大きく圧縮することができる。

F. タグの部分をタグ辞書105, 134の登録文字列と比較し、タグ文字列変換部111, 140において登録文字列と一致した部分を固定長バイトの置換文字(短縮文字列)に変換するので、タグ部分を大きく圧縮することができる。

【0345】G. XML文書の言語識別記号により、該当する言語の種別に応じた単語辞書を決定するので、多言語に対応しながらXML文書の復元が可能となる。

H. 伸長復元処理時には、置換文字比較部121, 153においてタグ内の置換文字をタグ辞書105, 134の登録置換文字と比較し、タグ文字列逆変換部122, 154において、登録置換文字と一致した置換文字をタグ内文字列に逆変換するので、タグを正確に復元することができる。

【0346】1. 伸長復元処理時には、置換文字比較部121, 153において平文内の単語番号(置換文字)を各言語の登録単語番号と比較し、単語番号逆変換部123, 155において、登録単語番号と一致した単語番号を単語文字列に逆変換するので、平文の単語番号を各言語に正確に復元することができる。

J. 可変長符号化部142により置換処理後の文字列(置換文字や単語番号)をさらに可変長符号化することで、XML文書の圧縮率をより高めることができる。

【0347】〔4〕その他

なお、本発明は上述した実施形態に限定されるものではなく、本発明の趣旨を逸脱しない範囲で種々変形して実施することができる。例えば、上述した実施形態では、構造化文書がXML文書である場合について説明したが、本発明は、これに限定されるものではなく、他の構造化文書、例えばHTML文書やSGML文書などにも同様に適用され、上述した実施形態と同様の作用効果を得ることができる。

【0348】

【発明の効果】以上詳述したように、本発明の構造化文書の圧縮方法（請求項1～12）および圧縮装置（請求項13～20）並びに構造化文書圧縮プログラムを記録したコンピュータ讀取可能な記録媒体（請求項21～28）によれば、以下のような効果ないし利点が得られる。

【0349】（1）文書実現値における要素の本構造を解析し、その解析結果に従って、文書実現値における葉要素についての情報を、親要素の属性としてこの親要素の開始タグ内に移すことで（請求項1、13、21）、より具体的には、葉要素についての開始タグ、終了タグおよび内容を文書実現値から削除し、葉要素についての情報である要素名および内容を、それぞれ親要素の属性名および属性値として親要素の開始タグ内に付加することで（請求項2、14、22）、葉要素にかかる記述を親要素の属性として取り扱うことができ、葉要素の開始タグや終了タグを記述する必要がなくなり、構造化文書の特徴を損なうことなく、また、検索可能な状態に保持したまま、葉要素にかかるタグの記述が省略・圧縮される。

【0350】従って、構造化文書の圧縮率を大幅に高めることができ、ひいては、大規模なデータベースを取り扱うシステムにおいて文書データの格納効率を大幅に高めることができ。特に、多数の短い語句（内容）を挟んだ開始タグと終了タグとの対表現を省略することができるので、その圧縮率を大幅に高めることができる。

【0351】（2）葉要素の開始タグ内に属性が記述されている場合、属性にかかる属性名および属性値を、それぞれ親要素の属性名および属性値として親要素の開始タグ内に付加することで（請求項3、15、23）、葉要素の属性にかかる記述も親要素の属性として取り扱われ、構造化文書の圧縮率をより高めることができる。

（3）親要素の終了タグを削除するとともに親要素の開始タグを空要素タグに変更することにより（請求項4、16、24）、さらに親要素の終了タグを構造化文書の記述から削除することができ、構造化文書の圧縮率をより高めることができる。

【0352】（4）文書型定義における要素の本構造を解析し、その解析結果に従って、葉要素についての情報を、文書型定義から削除し親要素の属性として文書型定義で再定義することで（請求項5、17、25）、より具体的には、葉要素の要素型宣言を文書型定義から削除するとともに葉要素にかかる記述を親要素の要素型宣言から削除し、その葉要素の要素型宣言にかかる情報を親要素の属性として再定義することで（請求項6）、文書実現値に対して行なわれた圧縮に対応した圧縮処理が文書型定義に対しても行なわれ、葉要素にかかる記述を親

要素の属性として取り扱うことができる。従って、構造化文書の特徴を損なうことなく、また、検索可能な状態に保持したまま、葉要素にかかる要素型宣言の記述が省略されて文書型定義が圧縮され、構造化文書の圧縮率をより高めることができる。

【0353】（5）文書型定義で葉要素の属性が葉要素の属性リスト宣言により定義されている場合、葉要素の属性リスト宣言を文書型定義から削除し、その葉要素の属性を親要素の属性として再定義することで（請求項7）、葉要素の属性にかかる記述も親要素の属性として取り扱うことができる。従って、構造化文書の特徴を損なうことなく、また、検索可能な状態に保持したまま、葉要素にかかる属性リスト宣言の記述が省略されて文書型定義がより圧縮され、構造化文書の圧縮率をより高めることができる。

【0354】（6）文書実現値のタグ内の記述を解析し、その解析結果に従ってタグ辞書を作成し、そのタグ辞書を用いて、文書実現値のタグ内に記述された文字列を、その文字列よりも短く且つその文字を特定しうる短縮文字列に置き換えることにより（請求項8、18、26）、構造化文書の特徴や構造を損なうことなくタグ内の文字列が圧縮されるので、構造化文書の圧縮率を大幅に高めることができ、ひいては、大規模なデータベースを取り扱うシステムにおいて文書データの格納効率を大幅に高めることができる。

【0355】（7）文書実現値のタグ内や文書型定義の記述を解析し、その解析結果に従ってタグ辞書を作成し、そのタグ辞書を用いて、文書実現値のタグ内や文書型定義に記述された文字列を、その文字列よりも短く且つその文字を特定しうる短縮文字列に置き換えることにより（請求項9、19、27）、構造化文書が文書型定義を有している場合であっても、構造化文書の特徴や構造を損なうことなく文書型定義の文字列が圧縮されるので、構造化文書の圧縮率を大幅に高めることができ、ひいては、大規模なデータベースを取り扱うシステムにおいて文書データの格納効率を大幅に高めることができ。

【0356】（8）タグ内や文書型定義に記述された要素名および属性名を短縮文字列に置き換えることにより（請求項10、20、28）、検索可能な状態に保持したままタグ部分や文書型定義を圧縮することができる。つまり、要素名および属性名について置換を行ない、属性値は元の形のまま保持することで、文書データを伸長することなく圧縮した状態のまま検索や文書構造の把握を行なえるようになっている。従って、文書圧縮後に文書の構造を解析して検索を行なう際に、圧縮された文書の伸長を行なう必要がなく、大規模なデータベースにおいて文書データを圧縮格納しても、その文書データの検索処理等を短時間で行なうことができる。

【0357】（9）単語辞書を用いて、文書実現値の内

容に含まれる単語文字列を、その単語文字列よりも短く且つその単語文字列を特定しうる短縮文字列に置き換えることにより（請求項11）、構造化文書の平文部分（文書実現値の内容）が圧縮されるので、構造化文書の圧縮率をさらに高めることができる。（10）置換処理後の文字列をさらに可変長符号化により圧縮することで（請求項12）、構造化文書の圧縮率をより高めることができる。

【図面の簡単な説明】

【図1】本発明の第1実施形態としての構造化文書の圧縮装置の機能構成を示すブロック図である。

【図2】（A）～（D）はいずれも第1実施形態における構造化文書（文書実現値）の圧縮原理を説明するための図である。

【図3】（A）～（C）はいずれも第1実施形態における構造化文書（文書実現値）の圧縮原理を説明するための図である。

【図4】（A）および（B）はいずれも第1実施形態における構造化文書（DTD）の圧縮原理を説明するための図である。

【図5】第1実施形態における構造化文書の圧縮手順を説明するためのフローチャートである。

【図6】第1実施形態における文書実現値解析手順を説明するためのフローチャートである。

【図7】第1実施形態における文書型定義解析手順を説明するためのフローチャートである。

【図8】第1実施形態における文書実現値構成変更手順を説明するためのフローチャートである。

【図9】第1実施形態における文書型定義構成変更手順を説明するためのフローチャートである。

【図10】（A）および（B）はいずれも第1実施形態による構造化文書（XML文書）の具体的な圧縮処理（第1例）を説明するための図である。

【図11】（A）および（B）はいずれも第1実施形態による構造化文書（XML文書）の具体的な圧縮処理（第2例）を説明するための図である。

【図12】（A）および（B）はいずれも第1実施形態による構造化文書（XML文書）の具体的な圧縮処理（第3例）を説明するための図である。

【図13】（A）～（D）はいずれも第1実施形態による構造化文書（XML文書）の具体的な圧縮処理（第4例）を説明するための図である。

【図14】本発明の第2実施形態としての構造化文書の圧縮装置の機能構成を示すブロック図である。

【図15】（A）～（D）はいずれも第2実施形態における構造化文書の圧縮原理を説明するための図である。

【図16】第2実施形態における構造化文書の圧縮手順を説明するためのフローチャートである。

【図17】第2実施形態における文書実現値文字列置換手順を説明するためのフローチャートである。

【図18】第2実施形態における文書型定義文字列置換手順を説明するためのフローチャートである。

【図19】第2実施形態における構造化文書の伸長手順を説明するためのフローチャートである。

【図20】（A）～（C）はいずれも第2実施形態による構造化文書（XML文書）の具体的な圧縮処理（第1例）を説明するための図である。

【図21】（A）～（C）はいずれも第2実施形態による構造化文書（XML文書）の具体的な圧縮処理（第2例）を説明するための図である。

【図22】（A）～（C）はいずれも第2実施形態による構造化文書（XML文書）の具体的な圧縮処理（第3例）を説明するための図である。

【図23】（A）～（D）はいずれも第2実施形態による構造化文書（XML文書）の具体的な圧縮処理（第4例）を説明するための図である。

【図24】（A）～（G）はいずれも第2実施形態による構造化文書（XML文書）の具体的な圧縮処理（第5例）を説明するための図である。

【図25】本発明の第3実施形態としての構造化文書の圧縮装置の機能構成を示すブロック図である。

【図26】本発明の第3実施形態としての構造化文書の伸長装置の機能構成を示すブロック図である。

【図27】第3実施形態におけるタグ辞書作成手順（タグ辞書登録手順）を説明するためのフローチャートである。

【図28】第3実施形態における構造化文書の圧縮手順を説明するためのフローチャートである。

【図29】第3実施形態における構造化文書の伸長手順を説明するためのフローチャートである。

【図30】（A）～（D）はいずれも第3実施形態による構造化文書（XML文書）の具体的な圧縮処理（第1例）を説明するための図である。

【図31】（A）～（G）はいずれも第3実施形態による構造化文書（XML文書）の具体的な圧縮処理（第2例）を説明するための図である。

【図32】（A）～（D）はいずれも第3実施形態による構造化文書（XML文書）の具体的な圧縮処理（第3例）を説明するための図である。

【図33】第3実施形態での構造化文書の圧縮手法を説明するための図である。

【図34】本発明の第3実施形態の変形例としての構造化文書の圧縮装置の機能構成を示すブロック図である。

【図35】本発明の第3実施形態の変形例としての構造化文書の伸長装置の機能構成を示すブロック図である。

【図36】第3実施形態の変形例における構造化文書の圧縮手順を説明するためのフローチャートである。

【図37】第4実施形態の変形例における構造化文書の伸長手順を説明するためのフローチャートである。

【図38】（A）～（C）はいずれも構造化文書（XM

文書)における一般的なタグの書き方を説明するための図である。

【図39】構造化文書(XML文書)のタグにおける一般的な属性の書き方を説明するための図である。

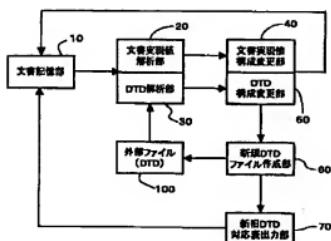
【図40】一般的なXMLプロセッサの処理について説明するための図である。

【符号の説明】

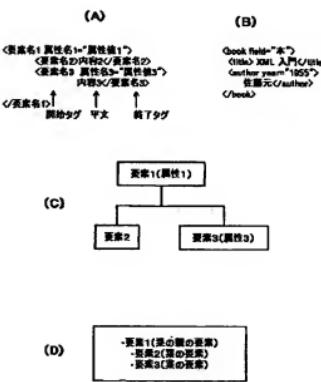
- 10 文書記憶部
- 20 文書実現値解析部
- 30 DTD解析部(文書型定義解析部)
- 40 文書実現値構成変換部
- 41 文書実現値文字列置換部
- 50 DTD構成変換部(文書型定義構成変換部)
- 51 DTD文字列置換部(文書型定義文字列置換部)
- 60 新規DTDファイル作成部
- 70 新旧DTD対応表示出力部
- 80 タグ辞書作成部
- 90 タグ辞書
- 100 外部ファイル
- 101, 124, 131, 156 文書記憶部
- 102 DTD条件調査部
- 103, 132 タグ平文識別部(文書実現値解析部)

- 104, 133 タグ文字列登録部(文書実現値解析部, 文書型定義解析部, タグ辞書作成部)
- 105, 134 タグ辞書
- 106, 135 文字列比較部(文書実現値文字列置換部, 文書型定義文字列置換部)
- 107, 136 言語識別部
- 108, 137 日本語辞書(単語辞書)
- 109, 138 中国語辞書(単語辞書)
- 110, 139 英語辞書(単語辞書)
- 111, 140 タグ文字列変換部(文書実現値文字列置換部, 文書型定義文字列置換部)
- 112, 141 単語番号変換部(単語文字列置換部)
- 113 単語番号ファイル
- 114 DTD記入部
- 120 単語番号ファイル記憶部
- 121, 153 置換文字比較部
- 122, 154 タグ文字列逆変換部
- 123, 155 単語番号逆変換部
- 125 旧DTD記入部
- 142 可変長字符串化部
- 143, 151 圧縮ファイル記憶部
- 152 可変長復号化部

【図1】



【図2】



【図3】

(A) <要素名1 属性名="属性値1" 要素名2="内容2" 要素名3="内容3" 属性名3="属性値3"/>

(B) <book field="本" title="XML 入門" author="佐藤元" year="1955"/> ↑
←空要素

(C) 要素1(属性1, 要素2, 要素3, 属性3)

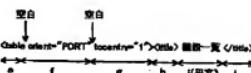
【図4】

```

1  <!DOCTYPE book [
2  <!ELEMENT book (title,author)>
3  <!ELEMENT title (#PCDATA)>
4  <!ELEMENT author (#PCDATA)>
5  <!ATTLIST book field (本 | 紙 | 小冊子) "本"
6  <!ATTLIST author year CDATA>
7  >

```

【図3.3】



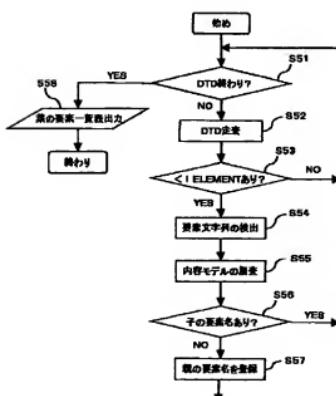
←要素名 属性名1="属性値1" 属性名2="属性値2".....→

```

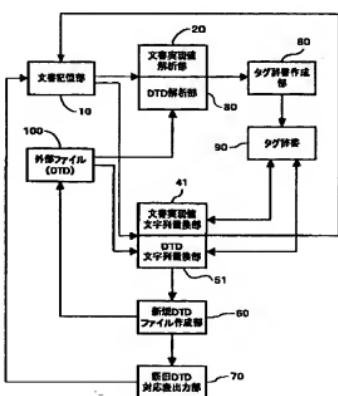
1  <!DOCTYPE book [
2  <!ELEMENT book >
3  <!ATTLIST book field (本 | 紙 | 小冊子) "本"
4  title #PCDATA
5  author #PCDATA
6  year CDATA>
7  >

```

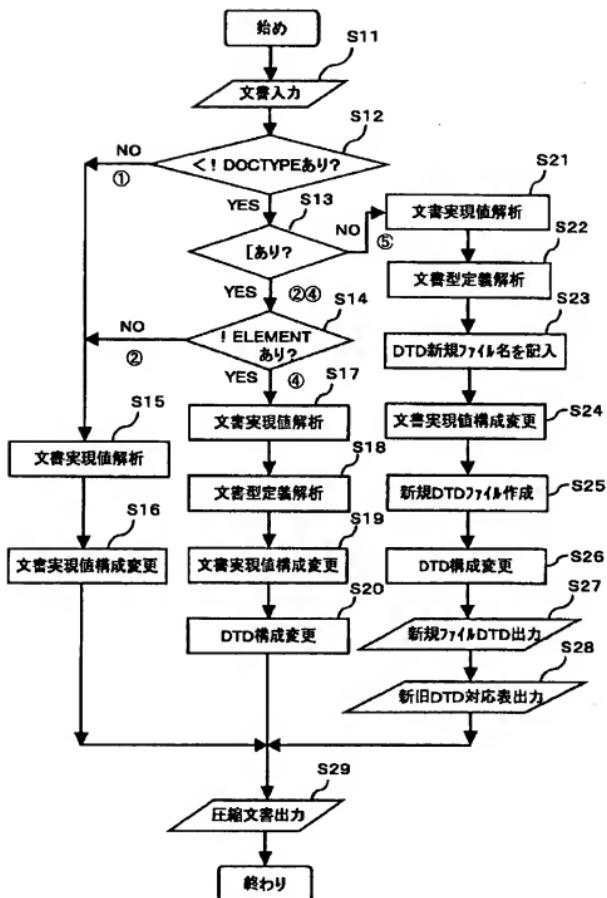
【図7】



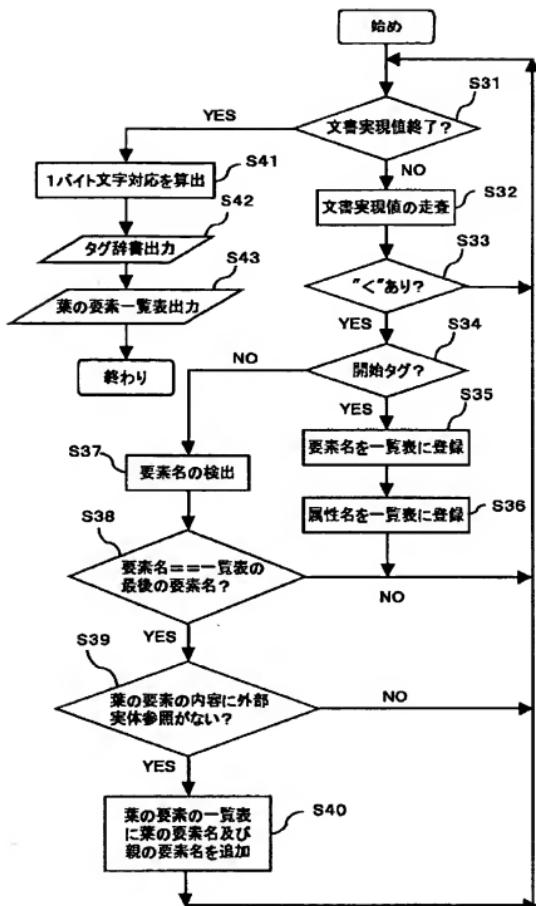
【図14】



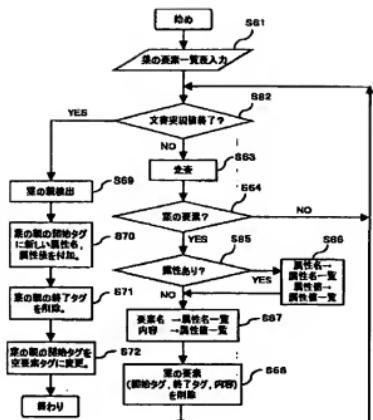
【图5】



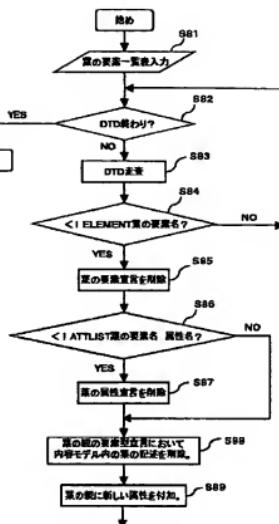
【図6】



〔图8〕



【図9】



【図10】

```
(A) 1 <?XML version="1.0"?>
2 <book>
3 <title> XML 入門</title>
4 <author year="1955">佐藤元</author>
5 </book>
```

(B) 1 <?XML version="1.0"?>
2 <book title="XML入門" author="佐藤元" year="1955"/>

【図11】

```

1  <?XML version="1.0"?>
2  <!DOCTYPE book[
3  <!ENTITY XML "Extensible Markup Language">
4  >
5  <book>
6  <title> XML(&XML;の略称)入門</title>
7  <author year="1955">佐藤元</author>
8  </book>

```



```

1  <?XML version="1.0"?>
2  <!DOCTYPE book[
3  <!ENTITY XML "Extensible Markup Language">
4  >
5  <book title="XML(&XML;の略称)入門" author="佐藤元" year="1955"/>

```

【図12】

```

1  <?XML version="1.0"?>
2  <!DOCTYPE book [
3  <ELEMENT book (title,author)>
4  <ELEMENT title (#PCDATA)>
5  <ELEMENT author (#PCDATA)>
6  <ATTLIST book field (本 | 雑誌 | 小冊子) "本">
7  <ATTLIST author year CDATA>
8  >
9  <book field="本">
10 <title> XML 入門</title>
11 <author year="1955">佐藤元</author>
12 </book>

```



```

1  <?XML version="1.0"?>
2  <!DOCTYPE book [
3  <ELEMENT book>
4  <ATTLIST book field (本 | 雑誌 | 小冊子) "本"
5          title  #PCDATA
6          author #PCDATA
7          year   CDATA>
8  >
9  <book field="本" title="XML入門" author="佐藤元" year="1955"/>

```

【図13】

(A)

```

1  <?XML version="1.0"?>
2  <!DOCTYPE book SYSTEM ".\book.dtd">
3  <book field="本">
4  <title> XML 入門</title>
5  <author year="1955">佐藤元</author>
6  </book>

```

(B)

```

1  <!ELEMENT book  (title,author)>
2  <!ELEMENT title  (#PCDATA)>
3  <!ELEMENT author (#PCDATA)>
4  <!ATTLIST book  field (本 | 雑誌 | 小冊子) "本">
5  <!ATTLIST author year CDATA>

```

(C)

```

1  <?XML version="1.0"?>
2  <!DOCTYPE book SYSTEM ".\book2.dtd">
3  <book field="本" title="XML入門" author="佐藤元" year="1955"/>

```

(D)

```

1  <!ELEMENT book>
2  <!ATTLIST book  field (本 | 雑誌 | 小冊子) "本"
3  title      #PCDATA
4  author     #PCDATA
5  year       CDATA>

```

【図15】

(A)

書き換え文字	変換名	属性名
	title	target="スクール CAI シリーズ NO. 563 301">

(B)

書き換え文字	変換名の文字列
■	title

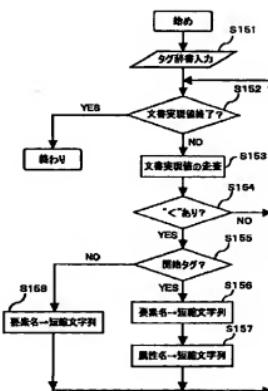
(C)

書き換え文字	変換名の文字列
A	target_

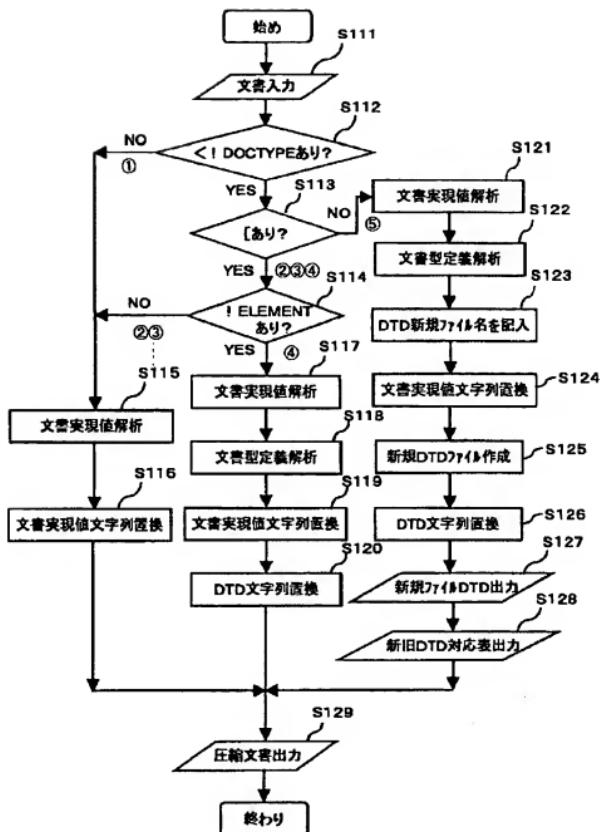
(D)

書き換え文字	変換名	属性名
■	A="スクール CAI シリーズ NO. 563 301">	

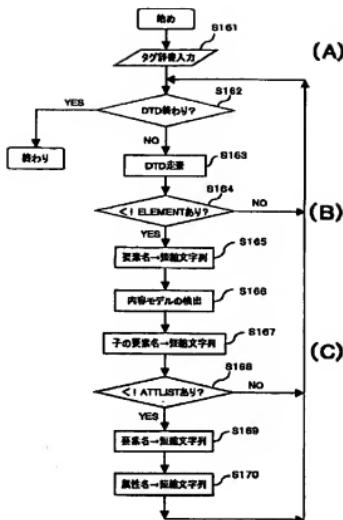
【図17】



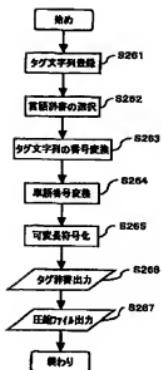
【図16】



【図18】



【図36】



【図20】

```

1 <?XML version="1.0"?>
2 <book>
3 <title> XML 入門</title>
4 <author> 佐藤元 </author>
5 </book>
  
```

```

1 <?XML version="1.0"?>
2 <a>
3 <b> XML 入門</b>
4 <c> 佐藤元 </c>
5 </a>
  
```

置換文字	要素名の文字列
a	Book
b	Title
c	Author

← 90

【図22】

```

1 <?XML version="1.0"?>
2 <!DOCTYPE book[
3 <!ENTITY para SYSTEM "http://www.xml.co.jp">
4 >
5 <book>
6 <title> XML 入門</title>
7 <author> 佐藤元 &para;</author>
8 </book>
  
```

```

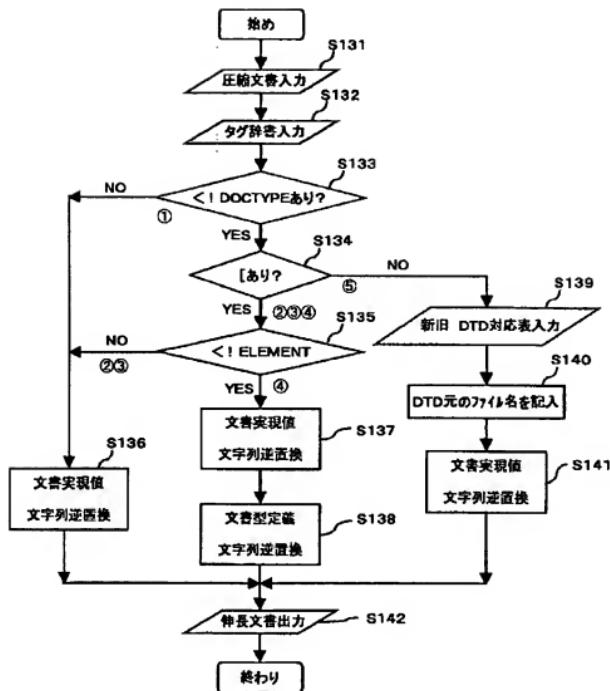
1 <?XML version="1.0"?>
2 <!DOCTYPE a[
3 <!ENTITY para SYSTEM "http://www.xml.co.jp">
4 >
5 <a>
6 <b> XML 入門</b>
7 <c> 佐藤元 &para;</c>
8 </a>
  
```

(C)

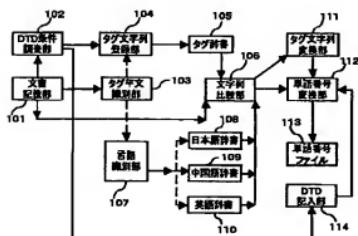
置換文字	要素名の文字列
a	Book
b	Title
c	Author

← 90

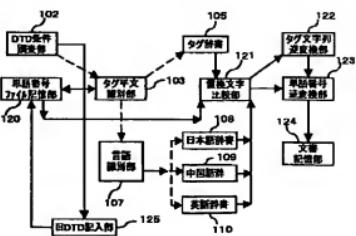
【図19】



【図25】



【図26】



【図21】

```

1 <?XML version="1.0"?>
2 <!DOCTYPE book[
3 <!ENTITY XML "Extensible Markup Language">
4 >>
5 <book>
6 <title> XML(&XML;の略称)入門</title>
7 <author> 佐藤元 </author>
8 </book>

```

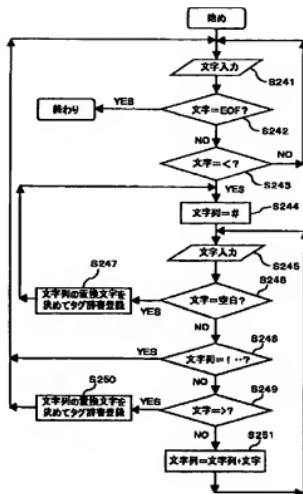
```

1 <?XML version="1.0"?>
2 <!DOCTYPE a[
3 <!ENTITY XML "Extensible Markup Language">
4 >>
5 <a>
6 <b> XML(&XML;の略称)入門</b>
7 <c> 佐藤元 </c>
8 </a>

```

置換文字	要素名の文字列
a	Book
b	Title
c	Author

【図27】



【図32】

(A) 压縮前のXML文書

```

1 <?XML version="1.0"?>
2 <book>
3 <title><book>XML の 基本</book>
4 <paragraph>XML は 之</paragraph>
5 </chapter>
6 </book>

```

(B) 压縮後のXML文書

```

1 <?XML version="1.0"?>
2 <book>
3 <title><book> XML </book>
4 <para><book> XML </book> </para>
5 </book>
6 </book>

```

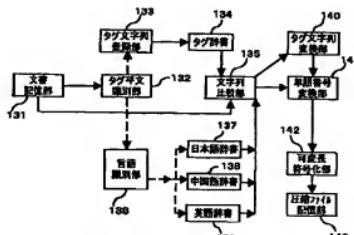
(C) タグ辞書の登録内容

置換文字	タグ名文字列
a	Book
b	Chapter
c	Title
d	Paragraph

(F) 日本語辞書の登録内容

置換番号	置換文字列
a	XML
β	の
γ	基本
δ	上記

【図34】



【図23】

```

1  <?XML version="1.0"?>
2  <!DOCTYPE book [
3  <!ELEMENT book (title,author)>
4  <!ELEMENT title (#PCDATA)>
5  <!ELEMENT author (#PCDATA)>
6  <!ATTLIST book field (本 | 雑誌 | 小冊子) "本">
7  >
8  <book field="本">
9  <title> XML 入門</title>
10 <author year="1955">佐藤元</author>
11 </book>

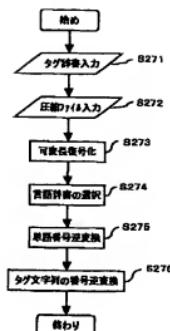
```

```

1  <?XML version="1.0"?>
2  <!DOCTYPE a [
3  <!ELEMENT a (b,c)>
4  <!ELEMENT b (#PCDATA)>
5  <!ELEMENT c (#PCDATA)>
6  <!ATTLIST a A(本 | 雑誌 | 小冊子) "本">
7  >
8  <a A="本">
9  <b> XML 入門</b>
10 <c>佐藤元</c>
11 </a>

```

【図37】



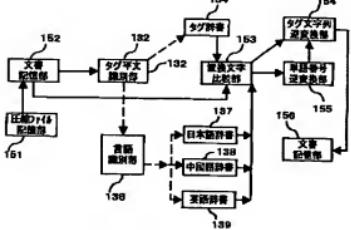
(C) 90

置換文字	要素名の文字列
a	Book
b	Title
c	Author

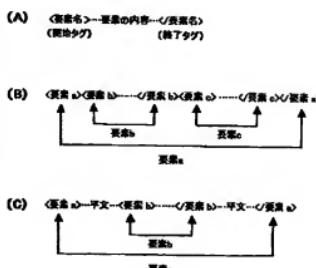
(D) 90

置換文字	属性名の文字列
A	Field

【図35】



【図38】



【図24】

(A)

```

1  <?XML version="1.0"?>
2  <!DOCTYPE book SYSTEM ".\book.dtd">
3  <book field="本">
4  <title> XML 入門</title>
5  <author>佐藤元</author>
6  </book>

```

(B)

```

1  <ELEMENT book  (title,author)>
2  <ELEMENT title  (#PCDATA)>
3  <ELEMENT author (#PCDATA)>
4  <ATTLIST book  field (本 | 雑誌 | 小冊子) "本">

```

(C)

```

1  <?XML version="1.0"?>
2  <!DOCTYPE a SYSTEM ".\book2.dtd">
3  <a A="本">
4  <b> XML 入門</b>
5  <c>佐藤元</c>
6  </a>

```

(D)

```

1  <ELEMENT a  (b,c)
2  <ELEMENT b  (#PCDATA)>
3  <ELEMENT c  (#PCDATA)>
4  <ATTLIST a A(本 | 雑誌 | 小冊子) "本">

```

(E)

置換文字	要素名の文字列
a	Book
b	Title
c	Author

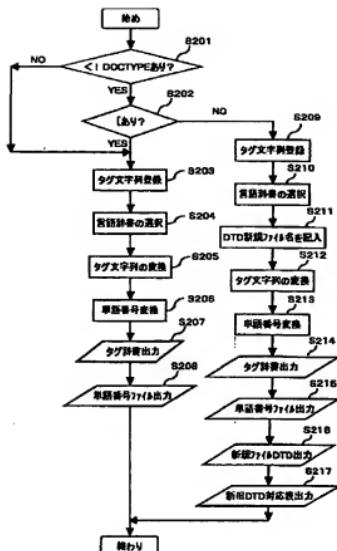
(F)

置換文字	属性名の文字列
A	Field

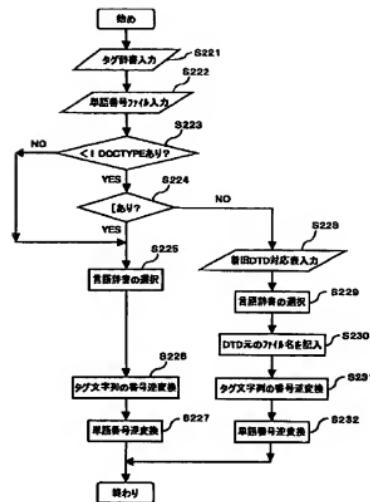
(G)

	新旧DTD
旧	Book. dtd
新	Book2. dtd

【図28】



【図29】



【図30】

(A) 実操作

```

1  XML version="1.0"?
2  DOCTYPE book [
3   ELEMENT book (chapter[])
4   ELEMENT chapter (title,paragraph[])
5   ELEMENT title (PCDATA)
6   ELEMENT paragraph (PCDATA)
7   ]
8   (book)
9   (chapter>章 XML の題名</title>
10  (paragraph>XML とは…</paragraph>
11  </chapter>
12  </book>
  
```

(B) 実操作

```

1  XML version="1.0"?
2  DOCTYPE book [
3   ELEMENT book (b)
4   ELEMENT b (c, d)
5   ELEMENT c (d)
6   ELEMENT d (PCDATA)
7   ]
8   (book)
9   (b>B T</b>
10  (c>C T--</c>
11  (d>D
12  </book>
  
```

(C) タグ登録の登録内容

登録文字	タグ内文字列
a	XML
b	Chapter
c	Title
d	Paragraph

(D) 日本語登録の登録内容

単語登録	単語登録
α	XML
β	の
γ	概要
δ	上記

【図31】

(A) 原始のXML文書

```

1  <XML version="1.0">
2  <?xml-stylesheet type="text/css" href="book.css"?
3  <Book>
4  <Chapter><title>XML の概要</title>
5  <graph>XMLとは…</p></graph>
6  </Book>

```

(B) XML文書をcssでcssize77:77; font-size:small;で表示

```

1  <ELEMENT book (title,graph)*>
2  <ELEMENT chapter (title,paragraph)*>
3  <ELEMENT title (#PCDATA)>
4  <ELEMENT paragraph (#PCDATA)>

```

(C) 原始のXML文書

```

1  <XML version="1.0">
2  <?xml-stylesheet type="text/css" href="book.css"?
3  <Book>
4  <Chapter> XML の概要
5  <graph> XML とは…</graph>
6  </Book>
7  <Book>

```

(D) 表示ファイル(book.css)を用いた表示

```

1  <ELEMENT book (title,graph)*>
2  <ELEMENT title (#PCDATA)>
3  <ELEMENT graph (#PCDATA)>

```

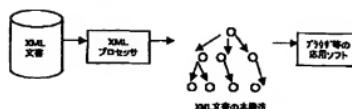
(E) タグ番号の登録内容

登録文字	タグ番号
a	Book
b	Chapter
c	Title
d	Paragraph

(F) 日本語登録の登録内容

登録番号	登録記号
a	XML
b	の
c	概要
d	とは…

【図40】



フロントページの続き

(51)Int.C1.7

識別記号

H03M 7/30

F I

H03M 7/30

チ-コド(参考)

Z

F ターム(参考) SB009 SA08

SB075 NR02 NR16

SB082 AA11 AA13 BA05 BA09 EA09

GA01 GC04

5J064 BA09 BA11 BA15 BC01 BC29

BD03